REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	February 1997	Final Report

4. TITLE AND SUBTITLE

The Self-Synchronous Schemotechnique as a Design Basis for The Fail-Safe Wafer Scale Integration Emergency Computers RPT #1

5. FUNDING NUMBERS

F6170896W0290

6. AUTHOR(S)

Prof. Adolf Filin

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Institute of Informatics Problems (IPI RAN)
Ul. Vavilova 30/6
Moscow 117900-GSP-1
Russia

8. PERFORMING ORGANIZATION REPORT NUMBER

N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

EOARD
PSC 802 BOX 14
FPO 09499-0200

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

SPC 96-4086

11. SUPPLEMENTARY NOTES

Report is in two parts: Intermediate Report No 1 and Concluding Report No 2

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

A

13. ABSTRACT (Maximum 200 words)

This report results from a contract tasking Institute of Informatics Problems (IPI RAN) as follows: The contractor will prepare a detailed technical report to include the analysis of present state 'rends in the development of computer systems IC base including a description of requirements for Emergency Computer schemotechnics. his report will also include a discussion of the main principles of self-timing as well as analysis and description.

14. SUBJECT TERMS

Computers

15. NUMBER OF PAGES

243

16. PRICE CODE

N/A

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

RUSSIAN ACADEMY OF SCIENCES

INSTITUTE OF INFORMATICS PROBLEMS
(IPI RAN)

SPECIAL CONTRACT PROGRAM

SPC 96-4086

THE SELF-SYNCHRONOUS SCHEMOTECHNIQUE AS A DESIGN BASIS FOR THE FAIL-SAFE WAFER SCALE INTEGRATION EMERGENCY COMPUTERS

INTERMEDIATE REPORT NO 1

WITH EUROPEAN OFFICE OF AEROSPACE RESEARCH AND DEVELOPMENT (EOARD)

CONTRACT F61708-96-W0290

MOSCOW, RUSSIA

FEBRUARY, 1997

PRINCIPAL INVESTIGATORS

Adolf V. Filin

Jury A. Stepchenkov

Leonid P. Plekhanov

Jury V. Rogdestvensky

Vladimir B. Egorov

Each Ph. D in Computer Science
(in computer architecture design and schemotechnics)

PREFACE

This intermediate report describes some scientific and practical outcomes obtained at the Institute of Informatics Problems of the Russian Academy of Sciences (*note*: the abbreviation IPI RAN is derivative from the Russian) under a fundamental research work on a new integral element base assigned for future real-time emergency computing systems and associated with the strictly self-synchronous schemotechnique. These computer systems are supposed to be embedded in process control systems of crucial (emergency) objects and to provide them continuous and safe real-time operation, extremal conditions included. The application areas for such systems might be: space and aviation, terrestrial and sea transport, industry (primarily nuclear, energetic, and chemistry), medicine, ecology, and others.

Main investigation objectives were nonconventional computer architectures and element base providing high-reliable parallel real-time computing. A natural combination of an original recurrent architecture with a strict approach to the self-synchronous schemotechnique is expected to meet most adequately the challenge of the wafer scale integration technology.

The report is concerned with:

- element base development prognosis for the high-reliable emergency computers (section 1)
- schemotechnique and element base of the wafer scale integrated circuits for the high-reliable emergency computers (section 2)
- implementation and element basis of the strictly self-synchronous circuits in the CMOS technology (section 3)
- CAD tools for the strictly self-synchronous circuit design (section 4).

A list of referenced literature is attached to the report.

Intermediate report contains 67 pages.

CONTENT

1 ELEMENT BASE DEVELOPMENT PROGNOSIS FOR HIGH-RELIABLE EMERGENCY COMPUTERS 1-1
1.1 Task statement 1-1
1.2 EB development prognosis 1-2
1.2.1 Integration scale 1-2
1.2.2 Switching rates 1-3
1.2.3 Production technologies 1-5
1.2.4 Technological effectiveness 1-5
1.2.5 Reliability 1-7
1.2.6 Functional specificity 1-8
1.2.7 Design techniques 1-9
1.2.8 Design automation 1-11
1.3 Brief conclusions 1-11

2 SCHEMOTECHNIQUE AND ELEMENT BASE OF WAFER SCALE INTEGRATED CIRCUITS FOR THE HIGH RELIABLE EMERGENCY COMPUTERS 2-1
2.1 Introduction 2-1
2.2 Basic functional requirements to the library elements of the WSI circuits 2-1
2.3 Choice of a schemotechnique for the WSI circuits 2-3
2.3.1 Arguments for the self-synchronous schemotechnique 2-3
2.3.2 Reasons preventing application and development of the self-synchronous schemotechnique 2-5
2.4 The key idea of the self-synchronization 2-8
2.5 Self-synchronous criteria 2-8
2.6 Design approaches to the self-synchronous circuits 2-9
2.7 Features of the self-synchronous schemotechnique with paraphase logic 2-11
2.7.1 Main differences of the self-synchronous schemotechnique 2-11

2.7.2 Features of self-synchronous circuits supporting the fault-tolerance 2-11
2.8 Hardware expenditures on implementation of the self-synchronous circuits 2-13
2.9 Comparison of synchronous, asynchronous, and self-synchronous principles of interaction 2-14
2.10 Quasi-self-synchronization 2-18
2.11 Importance of the self-synchronization principles for architectures of computer systems 2-20
2.12 Conclusion 2-21

3 IMPLEMENTATION AND ELEMENT BASIS OF THE STRICTLY SELF-SYNCHRONOUS CIRCUITS IN THE CMOS TECHNOLOGY 3-1
3.1 Technology for the self-synchronous ICs 3-1
3.2 The strict approach to the self-timing 3-1
3.3 Conditions of the self-synchronization at branching 3-2
3.4 Base elements of S^3 circuits 3-7
3.4.1 Structure and characteristics of S^3 base elements 3-7
3.4.2 Kinds and functions of the S^3 base elements 3-12
3.4.3 Use of the base elements in S^3 circuits implementation 3-15
3.5 Conclusions 3-16

4 CAD TOOLS FOR S^3 CIRCUIT DESIGN 4-1
4.1 General information about the FORCAGE system 4-1
4.1.1 Theoretical basis 4-1
4.1.2 Basic features 4-2
4.2 Brief overview of the subsystems 4-4
4.2.1 TRANAL subsystem of S^3 circuits analysis 4-4
4.2.2 TRAMIN subsystem of minimization of Boolean functions 4-5
4.2.3 TRASYN subsystem of S^3 circuits synthesis 4-6
4.2.4 TRACON subsystem of graphical input 4-8
4.2.5 System of S^3 element libraries on logic and behavioristic levels 4-8
4.3 Directions of development of the FORCAGE 3.0 system 4-9
4.4 Conclusions 4-10

LITERATURE

1 ELEMENT BASE DEVELOPMENT PROGNOSIS FOR HIGH-RELIABLE EMERGENCY COMPUTERS

1.1 Task statement

Among problems concerning the design of High-Reliable Emergency Computer (HREC) systems, the problem of creation of the element base (EB) for them is underlying, since just the EB appears the main delimiter of the computer generations.

With the integration scale growing, implementation of the whole electronic portion of a computer on a big chip, a "superchip" or silicon wafer, becomes a dominating concept in the electronic industry. This concept is referred to as Wafer Scale Integration (WSI). No WSI computers are available yet, but the LSI production leaders are able to implement on a wafer parts of such a computer and affirm that the WSI computers will be feasible by 2000th year.

Observation of the computing history shows, however, that every transition to a next computer generation has required to re-inspect architecture principles of the computer design. Like the earlier introduced notion of "VLSI-oriented architectures", the notion of "WSI-oriented architectures" is to come. Thus, although the EB was always an evolution "promoter" in the computing domain, a next step in the technology should be supported by some new design approaches enabling to apply all potential advantages of a next integration level.

Besides, any new design approach is always a compromise between "what is desirable" and "what is feasible". Accordingly, the potential advantages featuring a next integration level ("feasible") are to be juxtaposed with the general requirements to the HREC systems ("desirable").

The high reliability assumes some different criteria and estimates of EB properties as compared, for example, with the conventional performance/cost ratio for general-purpose computers. Specifically, the EB of the high-reliable computers not only has to operate in less "comfortable" ambient conditions, but also is desired to localize any failures and restrict expansion of ambiguous data and depreciation of obtained outcomes.

1.2 EB development prognosis

Further, following parameters are estimated and forecast in respect to the EB of the HREC systems:

- integration scale
- switching rates
- production technologies
- technological effectiveness
- functional specificity
- reliability
- design techniques
- design automation.

1.2.1 Integration scale

The integration scale does not have an "official" definition. In practice, usually following classification is typical:

- integrated circuit IC to 10^3 transistors
- middle-scale integrated circuit MSI to 10^4 transistors
- large-scale integrated circuit LSI to 10^5 transistors
- very large-scale integrated circuit VLSI to 10^7 transistors.

VLSI is the level of present 4th generation computers. A next step, Ultra Large-Scale Integrated circuit (ULSI), will cover the range from 10^7 to 10^9 transistors per chip. The future Giga-Scale Integrated circuit (GSI) technology is supposed to overcome the edge of a billion of transistors [1].

The increase of the integration level is associated with two major directions:

a) reducing the size of topological components

b) enhancing the silicon area used.

Reduction of the topological component size is not unlimited. The growth rates of the packaging density decreased approximately twice every ten years. In order to pass the .5 μm topologic boundary, some fundamentally technology advancements regarding new isolation

[1] The Wafer-Scale Integrated circuit (WSI) technology should not be regarded in this sequence as determining the number of transistors per a wafer rather than a chip.

methods and lowering power supply levels have been required. With the transistor size approaching the crystal lattice pitch, the transistor itself seizes to exist. The limit is not determined exactly but is estimated near .1 μm. Below this edge, the electron wave nature and quantum effects are expected to be exposed to light. On rough estimates, the edge of .1 μm will be achieved at the end of 90th years. The GSI technology assumes passage of this barrier and assimilation of the so called *nanotechnology* on a different physical base. Although some first test devices are expected to appear at the end of 90th years, the nanotechnology is a matter of the future century.

For enhancement of the silicon area there are no visible fundamental limits. The existing restrictions are purely technological and economical. Nevertheless, the die size increasing rates are only twice every 6 years that is significantly lower comparing with other trends. At present, the production output for the die sizes above 1 cm^2 is too low. Optimistic expectations permit to hope for 4 cm^2 dies. If the "wafer approach" will have won, the further growth of the die sizes can be seized. Only transition to some vertical (3-dimention) structures, with the element size below .3 μm, and further to super-lattice and quantum technologies will revive interest to conventional die sizes.

The existing ways of increasing the reliability and production output is reduced basically to the structural (hardware) redundancy always resulting in some extra silicon areas. For example, a simple triplication assumes more than 3-fold silicon area (with respect to the 2-of-3 selection hardware). It is evident that, from the viewpoint of reduction of the silicon areas, some more efficient ways are necessary for correction of both production and processing faults. Most easily it is achieved in the bus and matrix structures, with multiple identical elements being capable to the dynamic reconfiguration.

It is still unclear neither the exact boundaries of miniaturization of the conventional structures nor which specific type of devices is to replace them. There is, however, no doubt that any essential progress in either direction, the second especially, will drastically impact the computer architectures and design techniques.

1.2.2 Switching rates

The switching rate of a transistor is a major factor of enhancement of the system throughput. The existing experience shows the contribution of the technology was always greater than of other contributors, e.g. architecture improvements, and dominant in the performance

enhancement. The clock frequencies grew up more than 20 times during 20 recent years (specifically up to 400 MHz in Digital's *Alfa*).

Unfortunately, the increase of the switching rates and clock frequencies provokes many additional troubles:

a) increase of the consumed and dissipated power

b) influence of capacities and inductivities of silicon wires

c) inadequacy of the IC packages

d) synchronization and timing difficulties inside VLSIs.

The power consumed by a gate lowered permanently, from .3 mW per gate at 1990 year to .05 mW at 1995 year, but did not keep pace with the integration level. As a result, with the integration level growing up, even the most power saving CMOS technology does not provide the power consumption consistent with abilities of IC packages to dissipate the heat. The necessity to lower the power supply levels became aware with transition beyond the .5 μm barrier. Beside the excessive power dissipation itself, some serious 2nd order effects consequent upon fundamental physical limitations make to degrade transistor characteristics at the conventional 5 volt level and confine the packaging density. The sequence of $3.3 \rightarrow 2.2 \rightarrow 1.5$ volts is forecast in the future corresponding to the enhancing density. Under the lowered voltages, the CMOS technology possesses a sufficient reserve to reply the GSI challenge.

The source material determines the limit frequency characteristics. (It is known, for example, that GaAs provides 5-fold higher frequencies, up to 30 GHz, as compared with the silicon.) The existing technologies are expected to assimilate the sub-nanosecond range at the end of the current century. However, the increase of the switching rates requires by now huge expenditures and makes to transit to new, more expensive than silicon, materials or compositions (e.g. silicon-on-sapphire).

The search for adequate packaging solutions and acceptable carriers is to be done for the GSI chips and WSI wafers. At present, the Multi-Chip Modules (MCMs) seem to have a best perspective. The main reasons to choose MCMs consist in their possibility to implement essential portion of interconnections within an MCM that not only reduces the number of external pins but also improves the frequency properties of a device. Silicon laminas with re-programmable interconnection arrays may be used as a substrate for mounting the chips. Some prognoses predict the role of the PC boards to be reduced to connecting MCMs mainly by power/ground wires and shared (interface) information busses.

The synchronization and timing difficulties inside VLSIs are considered in the subsection *Design techniques*.

1.2.3 Production technologies

At present, following base technological materials are being either used or under development for the VLSI applications:

- silicon of 1st and 2nd generations
- semiconductor compositions of elements of III and IV groups of the Periodic element system
- superconductive materials
- light-conductive materials (quartz glass and plastics)
- rare-earth magnetics.

A comparative analysis enables to conclude that, to 2000th year, the silicon will not give in its positions. It is a most probable pretender to the main role in creation of the GSI circuits.

Although there are test devices with unique properties based on the non-silicon materials (i.e. GaAs, InPh, garnet, superconductive semiconductors, and others), the silicon-based integral microelectronics is developing so dynamically that the actual question confronting the computer architects is rather "how to use available dozens of millions of transistors" than "how to obtain some extra million". The silicon remains a most accessible, inexpensive, easily processed material which possibilities are still not exhausted. Only GaAs ICs are expected to compete in very restricted domains requiring enhanced switching rates (e.g. timing circuits).

The advantages of the CMOS/BiCMOS technology among plenty other available silicon implemented "logics" are well known, and there hardly are any doubts of its perspectives. The BiCMOS technology may be surely regarded as a technology of the HREC EB.

1.2.4 Technological effectiveness

There are two ways to advance the technological effectiveness:

- to improve the technology
- to design "flaw-tolerant" circuits.

An analysis shows most of invalid ICs to have solitary defects. Hence, possibility to use ICs tolerant to the solitary faults is expected to be an efficient way for neutralizing the technological

defects and ascending the IC production output to economically acceptable levels despite some hardware redundancy assumed by the approach. The approach is approved for the memories of large capacities and may be extended to other IC types.

However, the evaluations of the redundancy required for an arbitrary logic are not consoling. For example, majorization on the transistor level results in 33-fold complication of hardware. Even most practicable methods to neutralize the solitary defects complicate a circuit $3 \div 4.5$ times. Thus, improvements of production output via re-programmable structures are not a trivial task. It is being successfully solved nevertheless not only for memory chips but also for logical ICs and VLSI devices with an array structure. It is expected in the future that the economically profitable production of high-reliable VLSIs will be reached on a basis of large chips and wafers owing to fault-tolerant design methods. At least, obvious is the fact that any HREC developments must take into account some fault-tolerant design methods from the very beginning stages.

Related to the fault-tolerance are issues on:

a) testability

b) checkability.

The testability is a painful point of the VLSIs. In spite of considerable efforts, the testing tools are not adequate to the requirements of the VLSI design and production. The discrepancy is likely to become more evident for the GSI and WSI levels. The only hope to overcome the growing difficulties in testing the complicated devices is associated at present with the embedded checking properties. Therefore, the checkability is supposed to be a necessary property of the complicated self-reconfigurable HREC systems.

In the past, the check features were usually added to a device after synthesis of the block/unit structure. Together with inadequacy of available checking methods, it resulted in divergence of the check properties with a real model of probable faults and did not provide efficiency and reliability of the checking. For the HREC systems, with the very high processing certainty required, the problem of operative checkability becomes most important.

The generalized previous experience stresses necessity to provide the checkability throughout all system levels starting from the EB in order to achieve required operative efficiency. From this point of view, the self-synchronous schemotechnique (see subsection *Design techniques*) possesses unique properties unachievable by other competitors.

1.2.5 Reliability

The general notion of reliability comprises two basic aspects:

a) dependability

b) noise immunity.

Despite of available dependability-oriented systems and significant amount of knowledge, the problem of providing the real dependability as a whole can not be considered as solved. The known solutions relate to separate constituents and do not encompass all system levels. The methods approved on the system level can not be expanded "mechanically" on the level of units and schemotechniques. The concept of a dependability-oriented HREC architecture has to rely on a systematic multi-level fault-tolerant organization in respect to thoroughly investigated categories of possible breakdowns. Classification of possible breakdowns, investigation of their essence and impacts on the system operation should enable to find new efficient approaches, possibly technology-dependent, that would permit to neutralize technological faults on the levels of technology and schemotechnique getting the upper levels rid of related troubles.

The same is true considering the noise immunity. Classification and investigation of possible malfunctions relies on separation and influence of the destabilizing factors (DFs). DFs are considered here as a population of conditions influencing the behavior of electronic equipment under exploitation. An HREC system must be fully tolerant to all DFs, external and internal. Both DF groups are well studied and classified. The internal DFs are consequent upon errors under development or technology deviations under production. The external DFs are caused by the environment and time: ambient conditions (temperature, humidity), application conditions (electromagnetic fields, vibrations and accelerations, radiation, etc.), and aging conditions (abrasion, deterioration).

The modern technologies provide breakdown rates ~ 10^{-7} h^{-1} and malfunction rates ~ 10^{-4} h^{-1}. It has been noted that the breakdown rates weakly depend on the IC integration level while the malfunction rates increase, specifically because of diminished component sizes and power voltages as well as higher operation frequencies. As a consequence, namely malfunctions should be expected in the future to become the main "headache" of the HREC designers.

If the practicable methods of periodic self-testing are able, to a certain extent, to localize and neutralize breakdowns, they are absolutely inapplicable to seldom malfunctions which can arise also during the self-tests. Only operative means that monitor a process being checked in the pace of executed operations would be able to solve the problem. Hence, the HREC modules must

possess an ability of hardware self-checking and self-correction to support operation continuity and correctness, while the checkability must be regarded as a most important feature of the HREC systems from both points of view: production effectiveness and operation reliability.

The general final goal of HREC systems application — full unserviceability during entire lifetime — might be achieved as a reasonable combination of elaborated methods assuring sufficient level of both dependability and noise immunity.

1.2.6 Functional specificity

The declared objective — an entire computer on a single chip or wafer — generates a "mirror" trend of functional specificity of each particular product. An entirely complete system becomes very specific, with its "circulation" being restricted and cost excessive.

One of solutions of the contradiction might be treated in a "geometrical" way: from a point — to a line, from a line — to a square, and so on to multi-dimensional cubes. As applied to our concern, it would mean complication of the computer structures: from separate computers (i.e. "points") to a multi-computer systems with a shared exchange bus (i.e. "lines"), orthogonal computer arrays (i.e. "squares"), and multi-dimensional multi-computer systems (i.e. "cubes" and "hypercubes"). Multiplication of the system components, which number can reach dozens of thousands on the GSI level, is aimed at increasing both performance and reliability, since the multiple system components may be used for parallelization of computations as well as for reservation. The computational parallelization, in turn, enables either to speed up calculations or to save total consumed power, how strange it wouldn't seem, because the consumed power has a square-law dependency on the switching frequency.

An efficient system component multiplication within a HREC system has to meet following basic principles:

a) uniformity and regularity

b) hierarchical enclosure

c) explicit redundancy.

The uniformity of the system components will enable to facilitate both the design and application. The regularity is a sole way to provide interactions within a multi-system. It has been noted that, in implementations of conventional architectures, up to 80 % of chip area are occupied by interconnections rather than transistors. Hence, organization of interconnections between the

multiple uniform components is a most important problem of the system component multiplication.

The hierarchical enclosure will permit to preserve compatibility and succession of architecture solutions and simplify the design.

The explicit redundancy is an acknowledged way to provide enhanced reliability and viability.

Four classes of uniform structures may be considered as applicable to the HREC systems:

a) mono-bus linear structures

b) neighbor-connected arrays

c) algebraic graph networks

d) hierarchical structures.

The mono-bus linear structures, with a number of specific components (CPUs, memory banks, peripheral processors, DMA channels, etc.) connected to a shared common bus, are easiest and most trivial. The neighbor-connected arrays (matrices and cellular structures) are also well approved. The algebraic graph networks (*n*-cubic, link-and-recycle, shuffle-exchange, shift-and-replace, and cluster) are still experimental, and their applicability to general-purpose architectures is under question. Application of the hierarchical structures ("processor trees", "pyramidal computers") is also restricted at present to pattern recognition and signal processing. Therefore, research on new regular highly parallel structures consistent with both the general-purpose applications and GSI technology as well as development of general-purpose algorithms applicable to the parallel structures remain among the HREC challenges.

One of good solutions, though not exhaustive, is likely to introduce into a multi-system an inherent property of dynamic reconfigurations of its multi-system components. Such a property would be relevant not only for flexible accommodation of the structure to algorithmic features and drastic increase of the fault tolerance, but also should enhance the yield of valid chips/wafers under production.

1.2.7 Design techniques

At present, the LSI and VLSI designs apply almost exclusively the synchronous circuit design technique (further *synchronous schemotechnique*). The technique is based on a strict timing framework constituted by a single clock generator common for all parts of a system. The

synchronous technique is so simple and explicit that other alternatives would not be considered at all, shouldn't the technique appear to be inapplicable to very large systems.

The smaller is the area occupied on a chip by one transistor, the shorter is its switching time while the signal propagation time, clock signals included, is not changed. As a result, the growing relative clock deskews, within large chips, become compelling.

There are palliative solutions that move aside practicability frontiers for the synchronous schemotechnique. One of them, widely used now, is division of the system on separate *equichronous zones*, with the strict timing inside the zones and asynchronous interaction between them. However, this solution generates another problem of asynchronous arbitrating. The problem lies in transition from asynchronous interface signals to clocked signals inside the equichronous zones. Any synchronizer, a flip-flop usually, has a so called *metastable* undetermined state that arises accidentally at a random coinciding of appearance of an external signal to be clocked and active edge of an internal clock signal. The exit time from the metastable state is also a unpredictable random Poisson-distributed value. Probability of entering this state ascends with increase of clock frequencies (i.e. with growth of integration level). A theoretical analysis exposed to light the fundamental nature of the problem that does not have an absolute guaranteed solution.

This statement together with other drawbacks (the clock generation and distribution system consumes up to 25 % of the total power, generates undesirable noises, restricts scaling, etc.) makes obvious that:

- the synchronous schemotechnique, how attractive and approved wouldn't it be, is intrinsically incompatible with GSI and WSI technologies
- transition to *asynchronous* and *self-synchronous* schemotechniques [2] becomes inevitable.

The asynchronous schemotechnique entered the stage of practical test developments. The self-synchronous schemotechnique is at present more a theoretical concept. Nevertheless, namely this technique promises to implement most entirely all advantages of the GSI and WSI technologies.

[2] The self-synchronous schemotechnique is widely referred to as *self-timing*. We prefer the term *self-synchronization* because a strict approach we adhere to eliminates the notion of *timing* in circuits at all and makes the term *self-timing* merely senseless.

1.2.8 Design automation

The design of GSI and WSI devices insistently assumes some new approaches and design methods, a new *design methodology*. Development on all levels of: architecture, structure, logic, and topology — have to be combined as close as possible and submitted, to a great extent, to requirements of testability and checkability. All this could be implemented only by a new generation of "CAD-through" tools. Such tools, referred to also as "silicon compilers", are to support "solutions-through" of the design tasks from the very top ("strategic" tasks and system behavior) to the very bottom (EB and topological support).

It may seem that the necessity to develop entirely new HREC CAD tools incompatible with existing ones is a serious penalty, and it is. However, this objective necessity could facilitate to take this uneasy solution. It is very important for the self-synchronous schemotechnique which itself also requires some essentially new design methodology and new, incompatible with all existing, CAD tools. Thus, a new design methodology supported by new CAD-through tools is to combine innovations on all design levels from a whole system to its EB.

1.3 Brief conclusions

1. The progress in the EB of HREC systems will followed by considerable changes in:
   - architecture approaches
   - reliability provision methods
   - schemotechniques applied.
2. HREC systems are associated with the GSI and, especially, WSI technology levels.
3. The silicon-based (Bi)CMOS technology will preserve its positions as basic for HREC systems.
4. The noise immunity will be moved to the foreground of the reliability provision efforts.
5. The functional specificity will force to develop practicable methods of system reconfiguration and adaptation to specific application features.
6. The high reliability, noise immunity primarily, will be achieved due to:
   - essential redundancy of multiple uniform system components, with an extended dynamic system reconfiguration ability
   - testability and checkability inherent to all levels of an HREC system
   - special self-synchronous schemotechnique.

7. The accent in construction bases will be shifted from conventional PC boards to MCM-based silicon laminas.

8. New nonconventional CAD-through tools will encompass all design stages, with probable future extension to special HREC "silicon compilers" meeting requirements listed above.

2 SCHEMOTECHNIQUE AND ELEMENT BASE OF WAFER SCALE INTEGRATED CIRCUITS FOR THE HIGH RELIABLE EMERGENCY COMPUTERS

2.1 Introduction

Each digital EB is associated with some definite schemotechnique and specific tools of logical and electrical circuit design. The choice of a schemotechnique is very important for obtaining new quality characteristics of both EB itself and computer systems constructed on its base.

Below we describe the reasons for the choice of the *self-synchronous* (defined also as *asynchronous aperiodic*) schemotechnique as basic for creation of a new WSI level EB for computers assigned to be used in real-time applications with very high requirements to reliability.

The self-synchronous schemotechnique being chosen as a WSI circuits base for a new generation of real-time HREC systems enables to implement, in a natural way, a number of desirable and unique features which are practically unachievable at the conventional synchronous and asynchronous schemotechniques. The self-synchronous schemotechnique completely corresponds to the concept of the natural reliability [1] determining the HREC principles and architecture for computers being under development at the IPI RAN.

2.2 Basic functional requirements to the library elements of the WSI circuits

As it has been mentioned in section 1, a WSI implementation is expected to become real in nearest years. For creation of a WSI computer, it is necessary to design all its components (architecture, structure, EB, software, etc.) bearing in mind, from the very beginning, its future application areas (e.g. real-time systems) and possibility of its WSI implementation. This means, for the EB, the refusal from conventional sets of separate ICs. The schemotechnique of a WSI computer will consist of a set of functional elements (circuits) from which, on the stage of block synthesis, a general WSI circuit is assembled. Each functional element (FE), i.e. a functionally complete circuit, will be considered as a standard library element for multiple use.

In our case, the term *element* is understood as any real circuit fragment performing a definite logic function: inverters, various gates, flip-flops, decoders, ALUs, peripheral controllers, etc. The element may be applied as a typical constituent of more complicated circuits in operations of block synthesis.

Thus, the base library elements will underlie any schemotechnique.

As stated in section 1, any such an FE, regardless of its complexity, has to meet a certain set of qualitative requirements within an accepted implementation technology. With respect to this requirements set, each FE has to be:

a) developed in the CMOS technology;
b) equipped with an external self-synchronous interface;
c) assigned to operate at maximal possible rates defined by real component delays;
d) insensitive to deskews of signal edges and free of signal races and arbitrating problems;
e) continually self-checked (as a minimum) and self-corrected (as a maximum) regarding the most probable faults, so called *constant faults*, with the self-checkability being a natural and intrinsic FE property eliminating any "checkers-of-checkers" and any detected error terminating immediately the FE operation, if an operative self-correction is not provided;
f) ensured in respect to the constant faults, with all errors consequent upon constant faults being detected;
g) continually self-diagnosed, with faults being able to be localized and reported;
h) informationally fault-tolerant, with all outputs being locked on any detected error and the system being prevented thus from propagation of incorrect information;
i) insensitive to parametric faults, i.e. component parameter deviations caused by natural deterioration of materials, changes of temperature and power voltage;
j) insensitive to noises, with the noise immunity provided considerably higher versus the synchronous schemotechnique.

To fulfill all listed requirements, a reasonable amount of information redundancy is necessary to be introduced into the FEs.

As the functional complexity of circuits is not limited (a library can include functionally full units of all types: from a simplest self-synchronous inverter up to a complete self-synchronous computer), the hardware expenditures on implementation of the listed requirements in each FE have to be proved technologically and economically.

2.3 Choice of a schemotechnique for the WSI circuits

2.3.1 Arguments for the self-synchronous schemotechnique

Derivative from all said above, the self-synchronous CMOS schemotechnique seems most advisable for HREC implementations.

A comparison of the synchronous, asynchronous, and self-synchronous schemotechniques demonstrates advantages of the last one for implementation of the WSI circuits for the real-time computers. Theory of self-synchronous devices and the synthesis problems regarding the self-synchronous circuits and aperiodic devices are discussed in detail in [2, 3]. A number of inherent advantages of self-synchronous schemotechnique are also pointed out there.

The principle of the self-synchronization when a circuit (element, block, device, etc.) itself signals completion of a next operation phase is the main merit of the self-synchronous circuits comparing with the conventionally used. HREC development on the self-synchronization principles is advisable on many reasons, the most important of which are following:

1) Since the self-synchronous circuits do not require timing, a general timing scheme becomes unnecessary. A lot of problems inevitably accompanying implementations of synchronous computers are eliminated. Clock signal deskews and arbitration of competing signals are main causes of faults in the synchronous circuits. These troubles are naturally eliminated with the refusal from the general timing. The self-synchronous circuits are inherently free of signal competitions and the problem of reliable arbitrating.

2) The self-synchronous circuits operate on real delays (such an approach is referenced in the literature as "delay insensitive"). This feature results in:

- increase of operation speed
- immunity to parametric faults (unique feature not provided in the conventional schemotechniques) prolonging the lifetime of circuits and a computer as a whole
- lowering sensitivity to changes of temperature and extension of the range of permissible power voltage changes.

The self-synchronous circuits provide maximal speed possible within a manufacturing technology and specific conditions of application. Performance of the conventionally designed circuits determined by the most slow part. The self-synchronous circuits are free of such a limitation. They are synchronized by a fact of completion of operations in separate links.

The self-synchronous circuits are able to operate correctly within the range of parameters until active structures (transistors) function properly unlike the conventional circuits in which the range of correct operations is much more narrow because of possible concordance violations in separate parts.

3) A full self-checkability of the self-synchronous circuits is achieved regarding the constant faults that permits to consider the self-synchronous schemotechnique as fault-safe in respect to such faults and to obtain the following consequences:

- operation safety (operations are terminated at the moment a constant fault is detected confining thus propagation of wrong information)
- quick detection and localization of constant faults without testing
- more simple implementation of the operative self-repair that supports the real fault-tolerance.

4) . The technological requirements to the manufacturing of the WSI circuits become softer. As a result, the following advantages are provided:

- technology independence since races are eliminated
- increase of fitting ICs output
- simplified testing, especially effective for the WSI implementation.

The advantages (3) and (4) are consequent upon the maximal parameter independence and dependability and give direct economical effect.

5) A permanently steady load on the power supply units is achieved that lowers the level of noises on the power busses, reduces power consumption due to removing the permanently operating (as a rule, on highest frequencies) timing system, and eliminates peak-points of power consumption provoked, in the synchronous circuits, by many simultaneously switched flip-flops.

6) Possibility of gradual system development is preserved. Both types of circuits — synchronous and asynchronous — can be concurrently used in a system. The replacement (upgrade) of some devices in such a system does not urge re-design of the entire circuit as it were necessary in the case of synchronous circuits.

7) A possibility exists to verify the theoretical statements on the self-synchronization by implementation of a circuit in diverse EBs.

8) Some circuits unimplementable in the synchronous schemotechnique become feasible. Examples concern: asynchronous pipelines, fully self-checked checking hardware, etc.

Listed advantages of the self-synchronous circuits confirm that the self-synchronous schemotechnique satisfy all requirements to FEs, but (j), in subsection 2.2. This fact may be considered as a strong reason for selecting it as a basic HREC EB.

The requirement (j), subsection 2.2, has to be considered in more details. The problem of noise immunity of the self-synchronous circuits is not studied sufficiently up to now. At present, it is possible to state that their fault-tolerance (as well as fault-tolerance of the synchronous circuits) is provided on the physical level by technology, schemotechnique, shielding from external influences, and other physical methods. A logical approach to the design of the self-synchronous circuits adds nothing in respect to the increase of the fault-tolerance. However, the self-synchronous circuits are more safe regarding the internal noise generation as operating more smoothly being not paced by any external clocks. It is possible to hope that the self-synchronous schemotechnique is to provide indirectly to the WSI circuits more higher operation stability, as compared with the synchronous circuits, due to its insensitivity to the scatter of technology parameters and changes of power voltage, temperature, and humidity.

An explicit trend to enlarge the share of check functions may be revealed in the computer EB when passing to a next generation. Such a trend is also valid for large circuits. It is exposed first in creation of self-tested, then self-checked, and finally self-corrected circuits. Hence, it is possible to state that the quick quantitative changes in the VLSI technology have to be transformed in qualitative ones, and it seems that this transform will consist in integration of operation, check, and correction processes in a circuit, and result in creation of naturally reliable elements, units, devices, and computers as a whole. The checking process is to become an inherent processing constituent within a computer.

This forecast is the starting point for the selection of the schemotechnique and HREC EB design principles. The self-synchronous schemotechnique (with a related EB) corresponds to the concept of the natural harmony of processes and put the foundation of the highly reliable computers.

2.3.2 Reasons preventing application and development of the self-synchronous schemotechnique

Up to the present time, the self-synchronous circuits are not widely applied in the practical computer manufacturing. This is explained by both subjective and objective causes.

The main objective cause consists in a fact that this direction is absolutely new, recently arisen in the schemotechnique. For its development, all aspects — theory, methods, tools — have to be created from the very beginning. All other causes are secondary:

a) more complicated operation and design of the self-synchronous circuits;

b) a definite inertia of designers possessing a reach experience in the synchronous circuit design and approved powerful CAD systems; there are a lot of synchronous solutions widely used in practice and described in plenty books and manuals;

c) insufficiency of the theoretical basis of the self-synchronous circuits;

d) incorrect estimates of the hardware complexity of the self-synchronous circuits;

e) additional expenditures for embedding in the conventional circuits.

The cause (a) is eliminated by special CAD tools facilitating the development and checking circuits on semimodularity.

The causes (b) and (c) results from lacking any developed methods of dynamic analysis for the self-synchronous circuits that would guarantee correct operations of circuits within all range of possible delays of components and enable to estimate performance of the self-synchronous circuits on the design stage. It is known that the attempts to apply methods available for the synchronous circuits design to the dynamic analysis and synthesis of the self-synchronous circuits were not a success. Thus, creation of special tools for design and verification of the self-synchronous circuits is necessary [2]. Only considerable time and insistent effort in development of theory, popularization, and description of the self-synchronous design methods would solve this problem.

The origin of the cause (d) is a widely respected opinion that the self-synchronous circuits are excessively redundant. This opinion is based on comparisons of the conventional and self-synchronous circuits performing the same functions. Such comparisons are incorrect as juxtaposing circuits that are, in fact, not equivalent since the self-synchronous circuits possess some specific extra properties absolutely not proper to the conventional circuits. If two actually identical, with respect to reliability, checkability, etc., systems are compared then the result will another. From some level of complexity, the self-synchronous circuits become not only comparable but even less hardware (not to mention power) consuming.

The cause (e) interferes with the possibility to embed the self-synchronous circuits gradually. Combination of both conventional and self-synchronous schemotechniques is not well implementable. Only complex "overall" self-synchronous solutions on all system levels would give

good results. The final goal might be formulated as follows: from a self-synchronous information source — via a self-synchronous operation medium — to a self-synchronous information destination.

Besides, a special approach described in detail in the next section, the *strict self-synchronization*, facilitates one of most difficult problems of IC manufacturing: VLSI testing. Increased on a quadratic law versus the number of components, the expenditures on testing begin to prevail in total manufacturing expenditures. The suggested solutions reduce slightly the testing overheads on both design and exploitation stages. However, they treat consequences rather than the origin that lies in the binary logic of the synchronous systems free of redundancy. The tested circuits do not posses capabilities to neither detection nor correction. Only an entirely new approach to the schemotechnique with an intrinsic redundancy would solve this problem drastically. The self-synchronous schemotechnique seems to possess such an ability.

Below, positive qualities of the self-synchronous schemotechnique and interfering circumstances are juxtaposed.

Positive qualities

- Full self-checkability, with localization, on the constant faults
- Information safety of circuit operation, with the operation terminated, the outputs blocked, and the state indicated in the case of a fault
- No races at any delays
- Insensibility to parametric faults, changes of temperature and power voltage in a very wide range
- No external timing
- Lower power consumption and noise level
- Maximal speed due to operation on real component delays
- Increase of production output at manufacturing and higher technological stability
- Possibility of embedding into synchronous and asynchronous circuits

Interfering circumstances

- Higher complexity of circuits, more complicated understanding of the operation principles
- Insufficient propaganda of the self-synchronous circuits schemotechniques and lack of literature on the self-synchronous schemotechnique
- Redundant information representation with extra hardware and interconnections required
- Inadequately developed theory, especially concerning memory solutions
- Absence of commercial CAD tools, complete projects and computers

2.4 The key idea of the self-synchronization

Circuits of the modern synchronous VLSIs include on a chip a number of functionally complete subsystems (logic components) which concordant operation is provided by a centralized, common for all circuits, synchronization mechanism based on a stringent timing framework and impacting all components via physical clock signals. The difficulties connected with high quality timing signal sources and circuitries of their correct delivery to the target points are well known. Despite all tricks of designers and growing experience, a final solution of the problem is still not found.

The resolution of timing and signal arbitrating problems in the VLSIs lies in transition to a decentralized asynchronous aperiodic mechanism of components concordance that is referred to as the *self-synchronization*. In the self-synchronous VLSIs, therefore, the timing is determined by a natural sequence of internal events within the VLSI rather than by external clocks. An example of such an event is the moment of completion of a transition process determining the circuit is ready to interact with its environment. Thus, each self-synchronous system (the self-synchronous VLSI or any of its logic units) represents a set of self-synchronous elements having additional features to detect the moment of the transition process completion and an asynchronous interface to external environment (other elements). The indicator of transition process completion (further *indicator*) differs the self-synchronous circuits from synchronous and asynchronous ones. Development of the methods of detecting completion of transition processes and implementation ways of indicators are the key problems of synthesis of the self-synchronous circuits.

2.5 Self-synchronous criteria

In order to implement advantages of the self-synchronous schemotechnique in full volume, a circuit must follow strongly two criteria:

1. Each self-synchronous circuit must necessarily have, besides a required functional part, also a special hardware indicator signaling completion of transition processes.
2. Indicator implementation must be free of any hardware or software time settings.

Any self-synchronous circuit can be in two states (phases):

- *active* state
- *idle* state.

If a transition from one state to another has been initiated then a next initiation is permissible only after completion of all transition processes following the first initiation. Until the circuit has entered a new state and the indicator signals completion of the transition process, no other processes are permissible.

Schemotechnique solutions of indicators must not rely on hypotheses of any limitations on durations of transition processes (e.g. that signal propagation time through three consecutive gates is greater than through one gate of the same type).

2.6 Design approaches to the self-synchronous circuits

Two approaches to the self-synchronous circuits design exist: *logical* and *physical*. They differ by the detection methods of the transition process completion.

The logical approach uses a redundancy in the encoding of circuit inputs and outputs by special self-synchronous codes. Detection of some preliminary fixed codes by the indicator means completion of the active phase of circuit operation and, hence, completion of the transition process. The structure of a self-synchronous circuit using the logical approach is represented in *fig. 2.1*.

At the physical approach [4, 5], the level of current consumed by the functional part of the circuit from the power supply serves as a parameter checked in the indicator. Lowering the current level to a threshold minimum changes the state of indicator output that signals transition process completion in the functional part of the self-synchronous circuit. The structure of a self-synchronous circuit using the physical approach is represented in *fig. 2.2*.

Each of two approaches has its merits and drawbacks. The logical approach is more advanced and meets all requirements (a ÷ j) listed in subsection 2.2. Circuits designed on the logical approach look more natural, with information and control processes tightly combined in them. Faults in both functional and checking hardware are not distinguished, there is no necessity in special "checkers-of-checkers". The logical approach has been chosen for our WSI circuit schemotechnique, and all further considerations regarding the self-synchronous circuits are valid only for the logical approach.

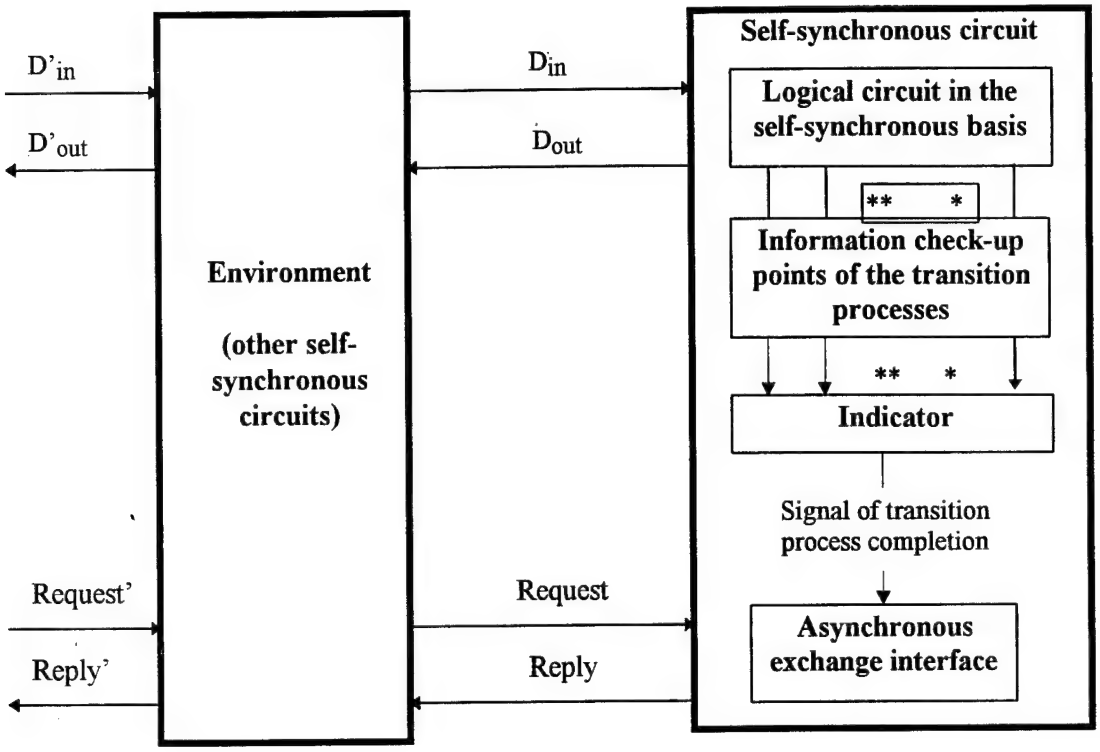


Fig. 2.1 **The Logical Approach To the Indicator**

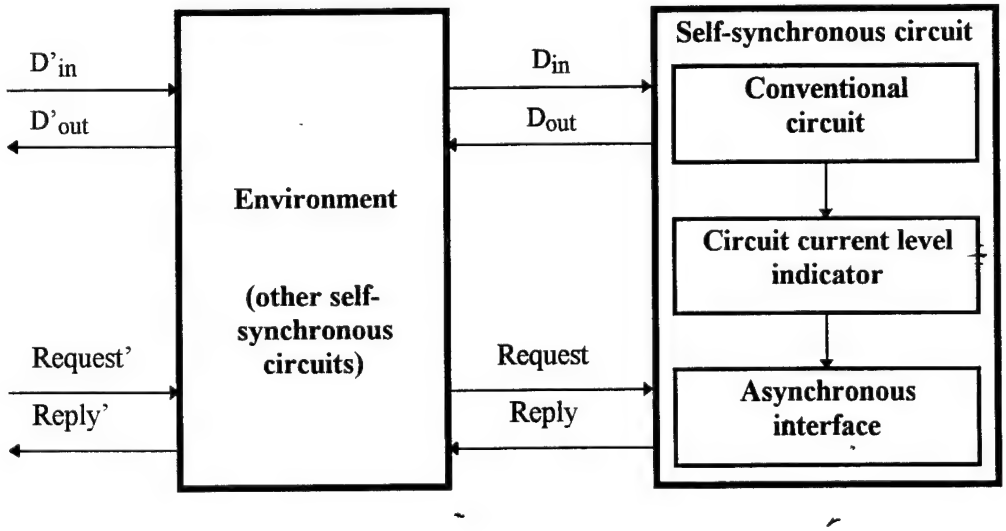


Fig. 2.2 **The Physical Approach To the Indicator**

The physical approach is more simple for implementation. With such an approach applied, the VLSI design can be accomplished by conventional methods. Decomposition of a VLSI logic scheme is performed, and each unit is equipped with an indicator consistent with individual unit parameters and interface logic identical for all units. The self-synchronous circuits designed on the physical approach operate faster, are less expensive in implementation (by 40÷50 %), and require less design time as compared with those using the logical approach. Unfortunately, the approach does not meet the reliability requirements, and monitoring the current level for units consisting of dozens of transistors is complicated. We neither apply this approach nor consider it further.

2.7 Features of the self-synchronous schemotechnique with paraphase logic

2.7.1 Main differences of the self-synchronous schemotechnique

The self-synchronous schemotechnique differs from the conventional one by following points:

1. Special self-synchronous encoding of information signals. The number of the encoded signals is always greater than the number of initial ones, i.e. hardware redundancy follows information redundancy. In the self-synchronous schemotechnique assigned for HREC applications, the paraphase coding is used. (The paraphase encoding assumes each logic variable being represented by two signals, inverse one to another, which are transferred on two separate wires. Doubling of connections should be taken into account under circuit development and routing.)
2. Two alternating phases under operation: *active* and *idle* (called also *spacer*). The phase alternation guarantees avoiding races. Phases are distinguished by the specific encoding of information signals, indicators, and, possibly, a special control signal.

2.7.2 Features of self-synchronous circuits supporting the fault-tolerance

It has been considered initially that the main advantage of the self-synchronous circuits comparing with conventional ones consists in:

- the speed increase due to operation on real element delays
- immunity to parameters deviation.

However, it has been established later that a not less important advantage of self-synchronous circuits is their ability to the self-checking. The ability to recognize the moment of transition process completion enables simultaneously a self-checking possibility on constant faults since, usually, an infinite element delay ("infinite 0" or "infinite 1" on an output) is a consequence of a constant fault.

In the theory of the self-checking, customary is a limitation by the constant faults only. Nevertheless, the developed diagnostics methods are able to encompass other fault types if the latters are reduced to the constant faults. The reduction methods are to be found and approved.

In the synchronous schemotechnique, the concept of the operative self-checking, i.e. detection of faults under process of normal operation, is based on such a division of sets of input and output states of a circuit in classes at which the reactions, on the same input states class, of the correctly and incorrectly functioning circuit relate to the different output states classes.

Although a redundant encoding enables to create self-checked synchronous circuits, the term *fully self-checked* is not entirely exact in respect to the synchronous circuits since the timing system is not checkable there. Unsuccessful attempts to create completely self-checked asynchronous circuits are explained by the fact that the problems of the self-checking and struggling against critical races were considered separately.

The following mechanisms provide detection of the constant faults in the self-synchronous circuits:

- encoding redundancy enables to distinguish erroneous states sets from correct sets
- phase alternation is simultaneously both a functional and checking aspect of operation.

The time redundancy following from the paraphase discipline is compensated by the possibility to operate on real delays of elements.

Appearance of an indicator informs about a correct completion of a current phase in a circuit element. Absence of the indicator during some critical time interval (indicator time-out) means there is a constant fault in the circuit element.

The self-checkability is a base for the "self-repairability". The latter, in turn, allows to ignore some VLSI manufacturing defects by reconfiguration as it has been discussed in section 1.

2.8 Hardware expenditures on implementation of the self-synchronous circuits

One of reasons restricting wide implementation of the self-synchronous schemotechnique in the design practice is an opinion that this schemotechnique is inherently too excessive. In fact, there are classes of devices for which the redundancy does not exceed 25 ÷ 40 % comparing with conventional synchronous solutions. In some cases, the self-synchronous schemotechnique permits even to reduce hardware expenditures. Only "formal" self-synchronous implementations of combinational circuits may result in doubling hardware.

The available comparisons are incorrect since circuits different from viewpoints of speed and reliability are compared. The self-synchronous circuits have several obvious merits that, of course, must be paid for. The picture is changed drastically when a comparative analysis concerns two equivalent circuits with identical characteristics of reliability, one being conventionally synchronous and another alternatively self-synchronous.

As an object of comparison, a "pure combinational" circuit could be taken, e.g. a centralized arbiter with radial connections. This circuit is chosen as supported by the existing self-synchronous CAD tools.

The self-synchronous radial arbiter meets requirements to the *Futurebus* protocol on the arbitrating procedure. Four versions of radial self-synchronous arbiters have been developed and investigated:

- v1 — conventional version
- v2 — version using base library elements in the disjunctive normal form with a *one*-spacer
- v3 — version (b) with different spacers (alternated *one*-spacer and *zero*-spacer)
- v4 — version (c) using base library elements in the disjunctive/conjunctive normal form.

Versions v2 ÷ v4 are fully self-synchronous and provide the self-checking on the constant faults. In the version v1, some additional checking hardware was added.

The hardware expenditures, in transistors, for all four compared versions are summed up in *table 2.1*.

Table 2.1 shows explicitly that:

1. The arbiter circuit in the disjunctive/conjunctive basis with alternated spacers, version v4, takes less hardware than other versions, conventional included, with preserving all self-synchronous advantages and providing higher speed.

2. Hardware ratios depend very little on the number of radial lines (i.e. arbitrated priorities).

Hardware Expenditures In the Arbiter Implementations								*Table 2.1*
Number of priority levels	3				6			
Circuit version	v1	v2	v3	v4	v1	v2	v3	v4
Number of transistors	474	646	606	432	888	1248	1190	812
Relative expenditures	1	1.36	1.28	.91	1	1.4	1.34	.91

2.9 Comparison of synchronous, asynchronous, and self-synchronous principles of interaction

The problem of events coordination and time concordance of operations in diverse parts of a computer system is one of most important. The commonly applied synchronous principle uses a clocking mechanism with strict limitations on all exchange phases regarding all interacting components:

- phase of connection (Pc)
- phase of exchange (Px)
- phase of disconnection (Pd).

As an example illustrating a synchronous exchange we represent in *fig. 2.3a* the timing diagram of information exchange between two devices: Master (initializer of the exchange) and Slave — using a bus type system interface (e.g. Multibus II) with the clock system {Ti}, in our case T1 and T2. The length of the clock signals T1 and T2 that determine the exchange speed in the implemented circuit have to be chosen for a worst case. The worst case means specifically maximum possible durations of element switching and a worst combination of their operational conditions (power voltage, temperature, length of wires, etc.). As a result, the synchronous circuits operate usually slower than they are able with respect to only parameters of specific elements in a real environment. The transition processes are not controlled.

In the case of information exchange on the asynchronous principle, one of the phase components, namely Px, becomes asynchronous (the length of the exchange phase is not limited externally) and is determined by the Slave performance. Probability of a wrong information exchange diminishes but not to zero as the limitations on the exchange duration and disconnection

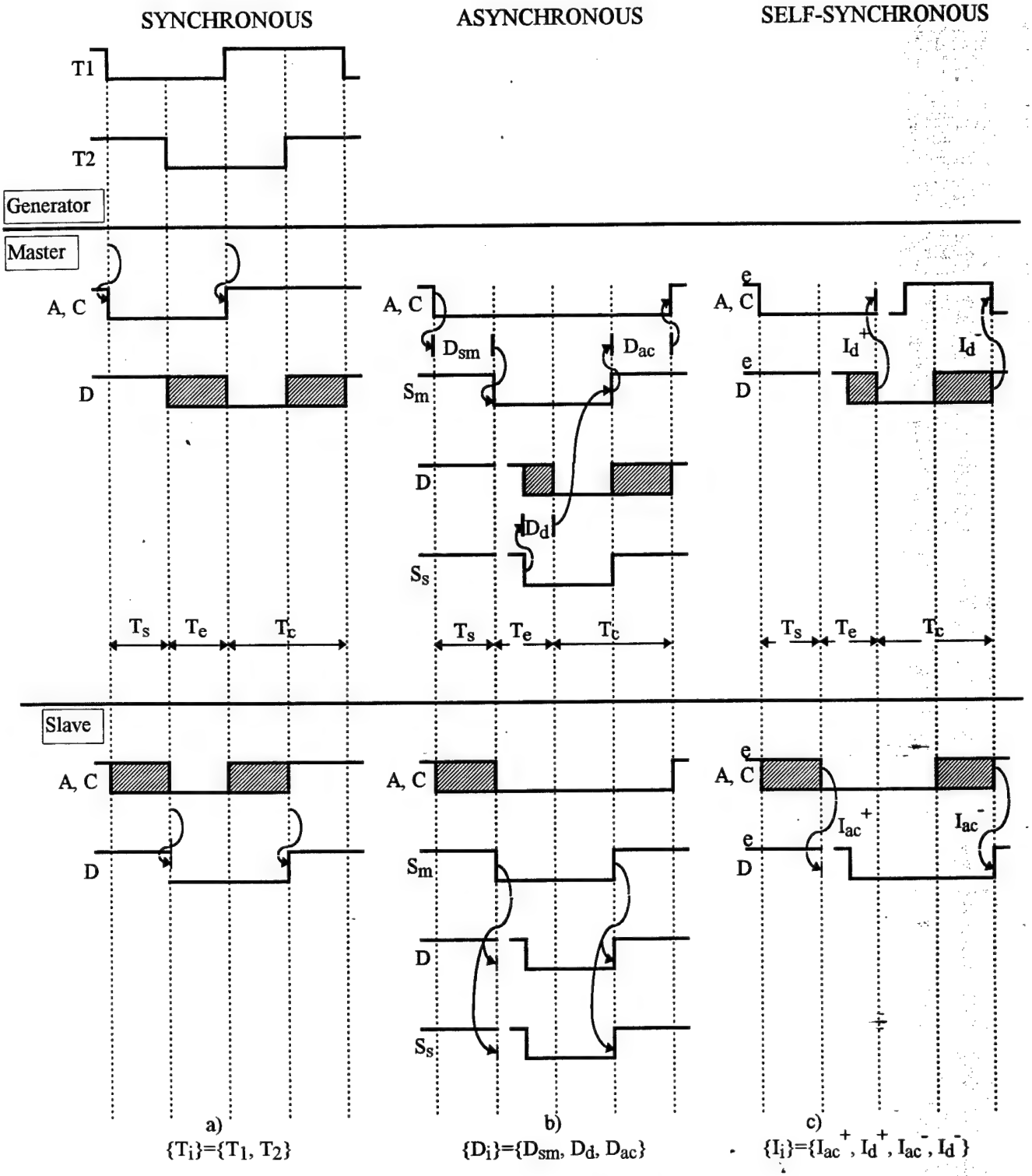
phase still remain (Pc and Pd are synchronous, i.e. not controlled, parameters. Thus, the asynchronous method uses a system of embedded delays {Di} for events coordination.

Absolute refusal from any limitations on duration of all exchange phases is achieved solely in the self-synchronous schemotechnique which can be considered as an alternative to both conventional approaches. In the self-synchronous schemotechnique, a system of clock generators ({Ti} — in the synchronous approach) or a system of fixed delays ({Di} — in the asynchronous approach) are replaced by a system of transition process completion indicators {Ii}. These indicators implement the mechanism of detection and fixing the moment of transition process completion in a self-synchronous circuit caused by a change on its inputs. *Fig. 2.3c* demonstrates that all phases in the self-synchronous exchange are not limited and are determined by the exchange partner reaction (i.e. a moment of transition processes completion). Duration of each exchange phase may be arbitrary but finite.

The example represented here shows the use of one of self-synchronous codes: paraphase code with a spacer — one of simplest for understanding and most universal (but not most efficient). Each bit transferred via the bus (of either address or data) uses two wires. The spacer is some service (nonoperational) set, e.g. state of bus lines "all ones" or "all zeros". The speed of the self-synchronous exchange is determined by the real delays of elements and, at the same other conditions, is higher than in the synchronous and asynchronous exchanges. The control of all transition processes in all self-synchronous exchange phases provides an upgraded level of information exchange reliability.

Thus, the asynchronous exchange principle used in the design practice is actually not asynchronous. The term "asynchronous" is valid only for one of three exchange phases, namely Px. Other exchange phases are, in fact, synchronous, but with a system of fixed delays being used instead of a clock system. For the self-synchronous exchange, the concept of *time* is excluded. All exchange phases are asynchronous and can be of arbitrary but finite length. From this point of view, the self-synchronous principle is a most "correct" and complete implementation of the asynchronous principle applied to all exchange phases.

Bearing in mind importance of this distinction, the signal graphs of information exchange between Master and Slave are given in *fig. 2.4* and *2.5*, on the asynchronous and self-synchronous principles, respectively. The signal graph is a convenient, compact, formally self-sufficient, and unambiguous tool for description of process interaction, in our case — for an exchange protocol on a common system bus.



$T=T_s + T_e + T_c$; A - address; C - control (read/write);

$S_{m(s)}$ - synchronization from master (slave); D - data; e - spacer

Fig. 2.3 **Data Exchange Modes Between Master-device And Slave-device**

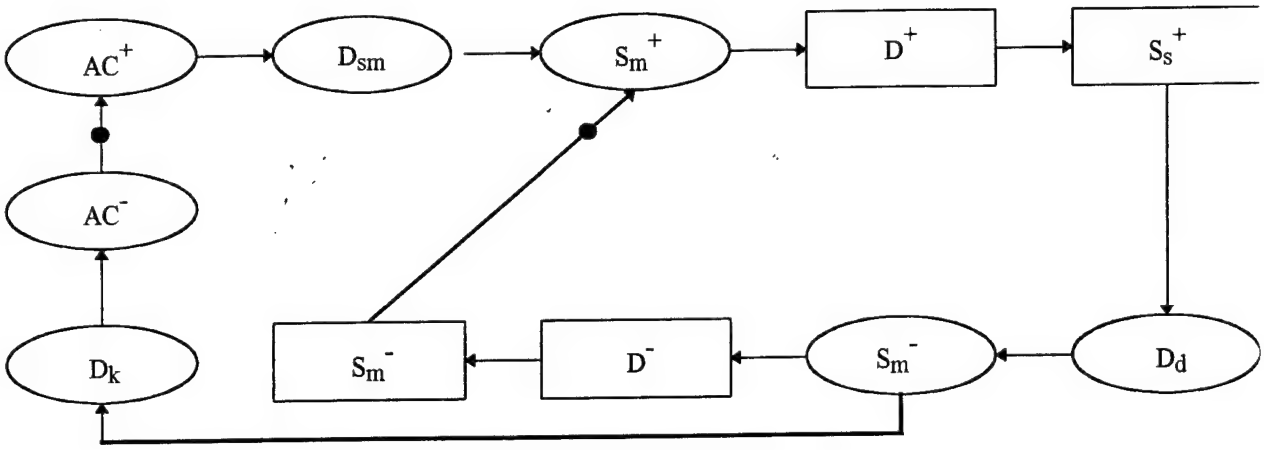


Fig. 2.4 **Change Diagram At the Asynchronous Approach**

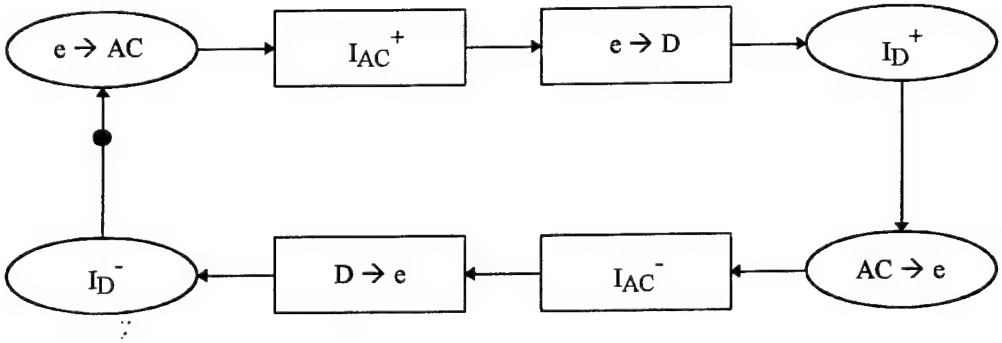


Fig. 2.5 **Change Diagram At the Self-synchronous Approach**

The signal graph is an *oriented marked monochromatic* graph (all its vertices are of the same type) in which each arc can be marked by arbitrary number of point-markers [6]. The rules of vertex-to-vertex transitions are analogous to those in the Petri networks. The arcs in the marked graph correspond to the *positions* of the Petri network. Each node of a marked graph corresponds to:

- change of the state of any simulated variable
- designation of embedded delay of some definite value compensating, for example, a signal propagation deskew on parallel wires
- "pure synchronizers" — additional vertices with no semantic interpretation.

The superscript "+" in the variable names denotes assertion while "–" — deassertion of the variable. The graph vertices denoted as $\mathbf{D_k}$ where **k** are embedded simulation delays of value of k (nanoseconds). The initial marking of the signal graph in *fig. 2.4* corresponds to such a state of connections when there are no information on address (*A*), control (*C*), and data (*D*) lines, the synchronization lines (*Sm* and *Ss*) are deasserted (the value of a corresponding variable is equal to zero), delays are off.

Fig. 2.5 shows a signal graph corresponding to the timing diagram in *fig. 2.3*. The transition $e \rightarrow AC\ (D)$ means a replacement of the spacer by a message defined by one of the self-synchronous code sets (paraphase); $AC\ (D) \rightarrow e$ is the reverse transition. *Fig. 2.5* demonstrates. that the exchange does not involve neither clocks nor fixed delays. The service indicator signals *Iac* and *Id* are formed and used within the interacting devices.

2.10 Quasi-self-synchronization

It is possible to implement in a full range all advantages of self-synchronous circuits only strictly following the Muller's hypothesis (see also section 3). One of positions of this hypothesis defines the requirements to the nature of delays: an element of a self-synchronous circuit is characterized by an inertial delay of arbitrary but finite value. This requirement permits to reduce "cost" of the self-synchronous implementation (see subsection 2.7), and, in practice, most part of designers develop circuits that are not really self-synchronous. Those are called quasi-self-synchronous circuits (efficient within some definite ratios of element delays). IPI RAN adheres the concept of a strict self-synchronization and deals exclusively with *Strict Self-Synchronous* circuits (S^3 circuits).

In practice, it is not reasonable to use always all possibilities of the self-synchronous circuits. For example, the self-synchronous circuits preserve their efficiency within practically unlimited range of component speeds. Let's suppose that, because of some external or internal factors, the speed of a circuit was decreased on an order of magnitude. Although the circuit itself remains efficient, would it be always acceptable from the viewpoint of the system performance ?

The introduction of agreements about "reasonable ratios" in delays is connected mainly with the limitations of the schemotechnique basis and with desire to come out of the Muller's hypothesis. In standard "gate array" circuits, usually, simple gates NOR and NAND are used, and

complex gates constructed from them do not meet the Muller's agreement with regards to the "indivisibility and non-inertiality" of a logic transformer.

Naturally, lowering the requirements to the delay ratios expands the class of circuits that can be considered as self-synchronous, with their implementation being usually simpler. But the question if the gain compensates the corresponding losses is still open. For example, refusal from the strict self-synchronous approach and transition to "reasonable ratios" means, in fact, refusal from the self-checking features of circuits (complete information regarding the completion of transition processes is not delivered to the circuit output). The faults diagnostics and fault-tolerance of such quasi-self-synchronous circuits becomes a more complicated problems than for the circuits with arbitrary delay ratios.

If a decision on the quasi-self-synchronous circuits design is taken then the agreement regarding the limited delay ratios have to be expanded also on the wires connected to the gate outputs and load capacitances. Thus, the delay ratios become dependent on the geometrical sizes of zone where a circuit is displaced and distances between components. In other words, such agreements are valid within the *equichronous zones* and may be used basically for synthesis of library modules while the interactions between equichronous zones have to obey the Muller's hypothesis.

Development of the quasi-self-synchronous circuits will require to use problem-oriented methods of their analysis and synthesis as methods of the strict synchronization are not applicable to them. Conventionally there exist a powerful set of methods for model description, analysis and synthesis of synchronous (or classic asynchronous) circuits, with all of their merits and drawbacks, based on the hypothesis of limited ("certificate") delay. It seems as more preferable to develop methods of analysis and synthesis for the quasi-self-synchronous circuits based on eventual models of types of change diagrams. Finding relations of strict precedence and simultaneity between signal changes enables to reduce implementation of limitations on delay ratios to manipulation with ordering changes of the signals. If the length (delay) of one of two simultaneous changes is less then of another then, at the same other conditions, it will be completed earlier, and this order may be reflected in the eventual specification of the circuit behavior. Such a simplification "cuts off" some of circuit states (by-passing some conflicting states) and enables to find critical delay ratios in an existing asynchronous circuit. No principle difficulties are seen on this way, but development of methods and algorithms requires considerable effort.

Considering the quasi-self-synchronous circuits as an intermediate stage on the way towards the strict self-synchronous circuits, IPI RAN has decided not to spend time and resources on development of their theory and design methods.

2.11 Importance of the self-synchronization principles for architectures of computer systems

Creation of HREC systems in the WSI technology is a complicated task that is impossible without a qualitative break through existing difficulties to special schemotechniques. Transition to the self-synchronous schemotechnique eliminates automatically some of these difficulties. In particular, the key problem for the WSI computer system implementation — synchronization and all related problems and troubles — is resolved automatically.

After transition to a submicron technology, the speed of circuits begins to be determined by delays in connecting wires rather than in active elements. And any technological differences in connecting wires implementation result in deskews in signal propagation. By Seitz [7], in the submicron WSI circuits the deskews may be hundred times greater of transition process duration in transistors. It makes practically impossible to preserve the relative switching rates achieved in the VLSI circuits at a classic synchronous approach. In a number of fundamental investigations, it was even stated entire impossibility to use the synchronous principle in the design of new circuits generation (WSI, GSI). The quick growth of the integration level aroused recently interest to the self-synchronization and its full-scale application to the computer design.

The research shows the implementation of the self-synchronization to be a productive theoretical and practical novelty which can be efficiently used on all levels of computer architectures, from the micro-level (transistor-topological) up to macro-level (system), with the integrity of the development concept being insured.

The self-synchronous schemotechnique eliminates the timing problems at all and opens a way to creation of WSI computers. It is an effective means to provide advanced reliability, maintainability, and fault tolerance of computer systems. At last, the self-synchronization delivers to computer systems features that are commonly applied in the nature and society:

- parallelism of processes (possibility to implement simultaneously a number of system events)
- asynchronous nature (removal of limitations on duration of system events that depend, in complicated systems, on a number of not controlled factors)

- system concordance (alternation of definite active and passive phases in a natural process flow on each level).

The self-synchronization is ideal for design of parallel high-performance high-reliable systems as providing a natural mechanism to control parallel asynchronous processes. This design principle is universal, applicable everywhere: from abstract synthesis to physical implementation.

2.12 Conclusion

1) Transition to the self-synchronous schemotechnique is an effective solution of some challenges in the synchronous systems.

With complexity and performance of digital computer systems growing, difficulties caused by global synchronization from an external clock source become compelling: deskews of signals in wires; increasing necessity and complexity of arbitration; negative influence of parameter unstability and pulse nature of loading on power supply devices; impossibility to fulfill requirements to the precision and stability of basic clock generators, etc. The observable stride to parallel asynchronous architectures in complicated VLSI systems for which an imposed global synchronization is unnatural requires appropriate schemotechnique support.

One of possible solutions is application of the self-synchronous schemotechnique to the HREC EB.

2) Transition to the self-synchronous schemotechnique in the WSI circuits is a warranty of information safety in the real-time HREC systems.

The self-synchronous schemotechnique and corresponding EB allow to obtain, in a natural way, a number of unique reliability features unachievable at the conventional schemotechniques. It also meets the requirements to the integral HREC EB.

Self-synchronous implementations of ICs of arbitrary complexity possess some features that bring forth a new idea in the reliability — guarantability, including:

- natural self-checking and self-diagnostics
- assured information fault-safety
- advanced parametric fault-tolerance
- maximum possible speed.

The listed features are very important for the emergency computer systems. They are automatically implemented in the self-synchronous circuits freeing thus designers from necessity

to solve these problems artificially. Composition of the listed features guarantees the only correct information to be processed in the computer system.

Such systems belong to a class of so called *guarantability systems* where the term "guarantability" is a functional characteristics which means assured provision of correct results despite possible faults, it is considered as a complex reliability characteristic of computer systems. The characteristics of readiness, safety, repairability, viability, and others partial quantitative characteristics are diverse manifestations of the same fundamental system feature — its guarantability.

3) Transition to the self-synchronous schemotechnique with global self-synchronous interactions in a system are a realty.

The analysis of features and characteristics of the self-synchronous schemotechnique shows that:

a) there are no serious reasons preventing its wide application in any system classes, it is reasonable to start nevertheless with the systems in which the new features of self-synchronous schemotechnique are especially important — emergency computers;

b) maximum advantages are obtained at a global application of the self-synchronous principle of interactions — from a source of information up to its receiver;

c) when the self-synchronous circuits are locally embedded into conventional digit systems, the effect may depend on the existing environment, and the final effect should be estimated in advance.

d) in order to bring the self-synchronization in practice of the computer systems design, the research and development effort is necessary to be intensified in the following areas:

- theory of design methods and algorithms for the strict self-synchronous WSI circuits as well as self-synchronous architectures for them
- parametrized libraries of the self-synchronous elements
- commercial CAD systems for strict self-synchronous WSI system architectures
- strict self-synchronous RAM and ROM elements.

3 IMPLEMENTATION AND ELEMENT BASIS OF THE STRICTLY SELF-SYNCHRONOUS CIRCUITS IN THE CMOS TECHNOLOGY

3.1 Technology for the self-synchronous ICs

It has been shown in section 1 that the CMOS/BiCMOS technology will underlie the EB of HREC systems due to following evident advantages:

- relative technology simplicity
- good integral performance
- low power consumption/dissipation.

3.2 The strict approach to the self-timing

Among several directions of research and development in the area of the self-timed circuits, we are developing a specific approach referred to as Strict Self-Synchronization (SSS or S^3).

Conventionally, the question of possibility to implement a self-timed or similar (i.e. not depending of delays) circuit is reduced to meeting a number of conditions on elements and their interconnections. In the related literature a logical description is considered in terms of delays in elements and wires. The elements are described as logical functions. The circuit implementation conditions are formulated as a ratio of delays in elements and wires. Such conventional estimates are completely insufficient in regard to the implementation possibility of the S^3 circuits.

The mathematical basis of the strict self-synchronization is the widely respected Muller's model [8]. The model comprises, explicitly or implicitly, a number of implementation assumptions which can be not satisfied in real circuits. These assumptions are divided in two groups concerning models elements and connections between them. An analysis has shown that violations of the model conditions can arise, for example, not only because of delays in wires but also because of deskews of input-output element characteristics. Not each element structure in transistors can be described by the Muller's model. Therefore, it makes necessary a special analysis, at the transistors level, of the electrical and time parameters of self-timed circuits and formulation of the implementation conditions.

Taking into account the first group of the assumptions allows to derive conditions of the Muller's model correctness for circuits with elements having different characteristics. Taking into account the second group enables to define a class of elements on which (and only on which) a correct implementation of the S^3 circuits is possible. These elements are further referred to as *base elements* of S^3 circuits or S^3 base elements.

3.3 Conditions of the self-synchronization at branching

Let's consider an example of a circuit having internal connections with branching, *fig. 3.1*.

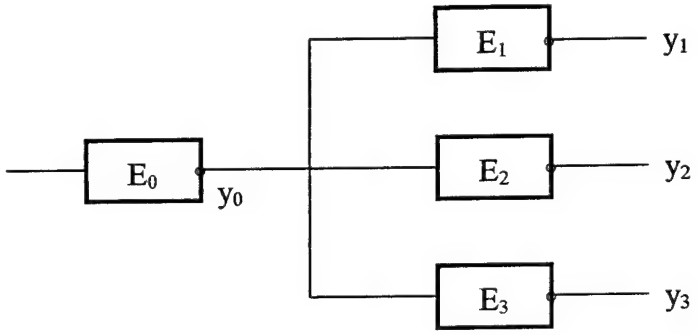


Fig. 3.1 **Example Of a Circuit With Branching**

Let the element $\mathbf{E_0}$ be *primary* and other elements ($\mathbf{E_1}$, $\mathbf{E_2}$, $\mathbf{E_3}$) — *secondary*. Let's also assume that switching of the secondary elements always follows switching of the primary element.

The Muller's model supposes implicitly that at the moment the primary (exciting) element switches internal transition processes in all secondary (excited) elements start simultaneously as shown in *fig. 3.2*.

In accordance with the Muller's model, duration of the excitation interval is, in fact, the element delay and can have an arbitrary finite value. In a real circuit, however, the excitation processes start not simultaneously that is consequent upon two main reasons.

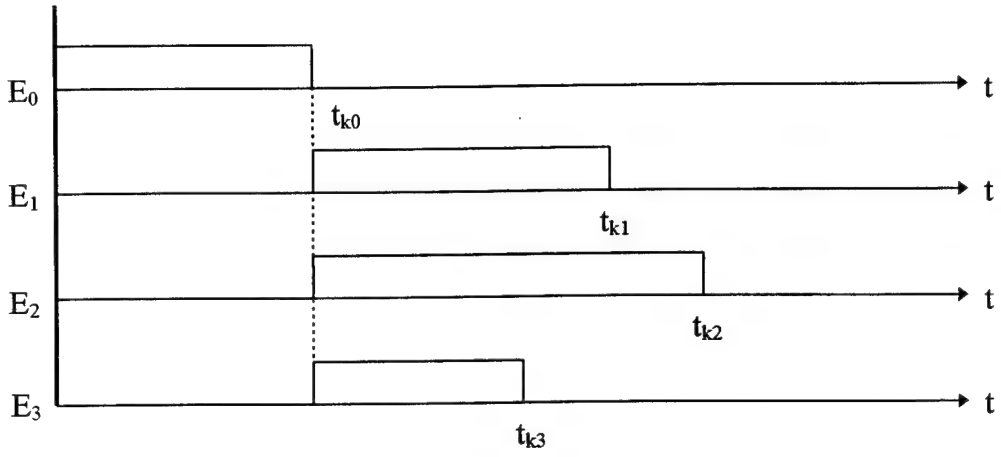


Fig. 3.2 **Excitation Processes In the Muller's Model**

1. The switching thresholds of excited elements are different. The difference of thresholds are explained by dissimilarity of the switching characteristics (SCs) of elements because of influence of the technological and electrical factors:
   - deskews of the elements characteristics at manufacturing
   - different influence on the SCs of temperature and power supply changes
   - SC dependency from the element loads
   - SC dissimilarity on diverse inputs.
2. The transport (pure) delay in branches (after the branching point) can differ considerably.

The listed reasons result in a real picture of excitation processes shown in *fig. 3.3*.

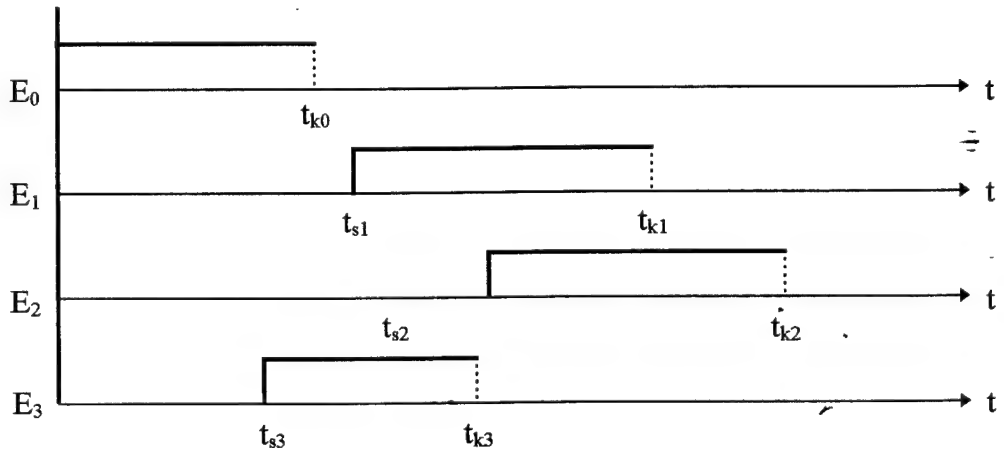


Fig. 3.3 **Excitation Processes In a Real Circuit**

By Muller, a moment of excitation completion for the primary element is the moment of excitation initiation for all secondary elements. Such a common moment does not exist in a real circuit. Therefore, moments t_{ki} (i = 0...3) in *fig. 3.3* are not defined.

Thus, a real circuit in *fig. 3.1* is not described immediately by the Muller's model. In order to achieve model adequacy, some additional superimposing limitations and delay reduction are necessary. Let's designate $t_{s\max} = \max_i t_{si}$ — a maximum initial moment of secondary elements excitation and assume:

$$t_{k0} = t_{s\max}. \tag{3.1}$$

We define thus the switching moment for a primary element as a moment when all secondary elements have been excited. All moments t_{ki}, (i = 1, 2,...) will be similarly defined in branchings after all secondary elements. Let's require meeting the conditions:

$$t_{ki} \geq t_{s\max}. \tag{3.2}$$

These conditions are necessary because otherwise functioning of the Muller's model will be broken, e.g. an element with maximal t_{si} had not been excited yet while an element for which the condition (3.2) is violated was already switched.

Let's define reduced delays for the circuit in *fig. 3.1* as

$$\tau_i = t_{ki} - t_{k0}. \tag{3.3}$$

These delays can be in the range $0 \leq \tau_i < \infty$.

As a result of the performed construction, following statement is true: a circuit with branchings may be described by the Muller's model in reduced delays, with the conditions (3.2) being validity conditions of this statement.

In many branching cases, true is the following inequality:

$$t = t_{s\max} - t_{s\min} << \tau_{li}, \tag{3.4}$$

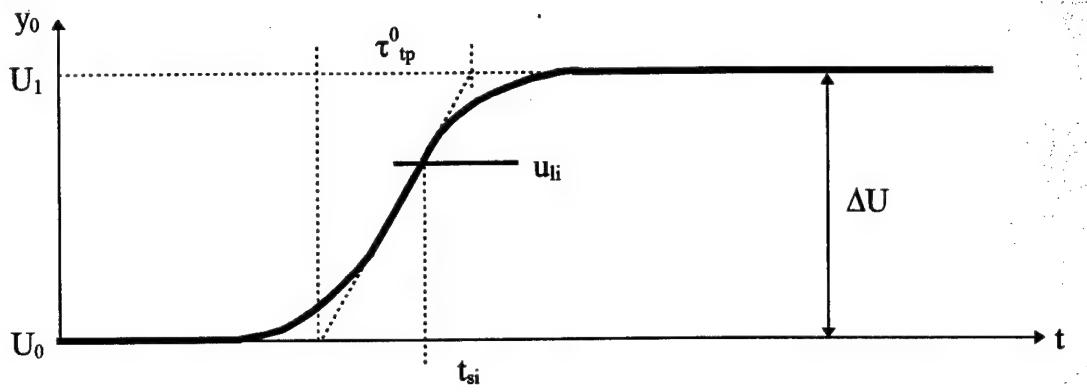
where $t_{s\min} = \min_i t_{si}$,

τ_{li} — threshold delay of i-th secondary element.

In such cases, the reduced delays practically coincide with threshold delays, and the conditions (3.2) are satisfied.

The conditions (3.2) may be violated in cases when the transition processes in one of secondary circuits are much faster than in others. Then, to meet conditions (3.2), it is possible to perform the delays leveling, for example, by augmenting the load on the "too fast" element, or some other way.

Let's consider in more details an important special case of absence of transport delays in wires. In this case, it is possible to obtain estimations for the conditions (3.2) via parameters of the library element and connections. *Fig. 3.4* shows the transition process for signal y_0.



t_{0tp} — transition duration;
ΔU — voltage difference between logical "high" (U_1) and "low" (U_0);
u_{li} — threshold level.

Fig. 3.4 **Transition Process For Signal y_0**

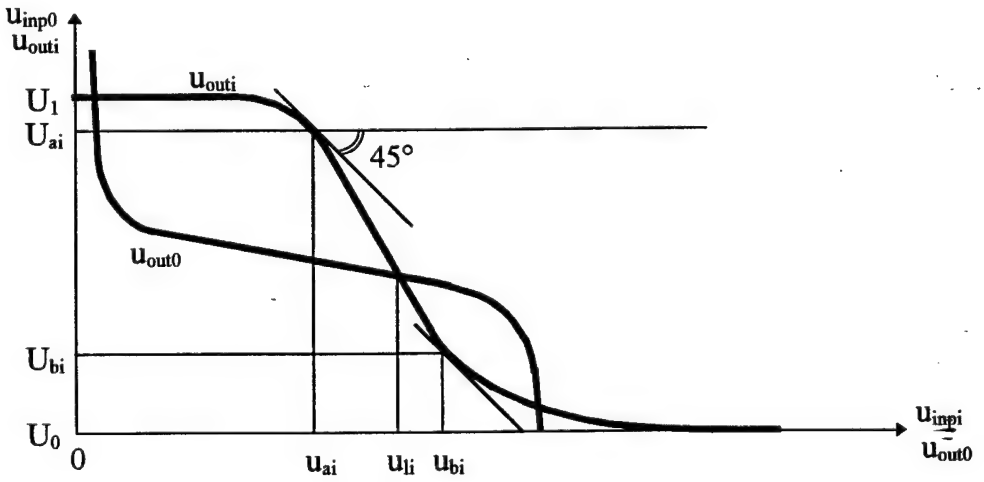


Fig. 3.5 **Combined SCs Of the Primary And i-th Secondary Elements**

Following condition must be satisfied for efficient elements:

$$u_{ai} < u_{li} < u_{bi}.$$

Let's estimate the deskew of initial excitation moments for the secondary elements t_{bi}:

$$\Delta t_{si} = 2t_{t0}\Delta u_{li}/\Delta U < 2\tau_{t0}(u_{bi} - u_{ai})/(U_{ai} - U_{bi}). \qquad (3.6)$$

The value $K_{si} = \left(U_{bi} - U_{ai}\right)/\left(u_{bi} - u_{ai}\right)$ is, by definition, the nonlinearity (steep) coefficient for i-th element. Its value reaches some decades for the CMOS technology.

The propagation delay may be represented as

$$\tau_t = \tau_{in} + \lambda C, \tag{3.7}$$

where: τ_{in} — internal delay independent of the load;

C — load capacity

λ — load delay factor.

The estimation (3.6) will be transformed to:

$$t_{si} < 2\left(\tau_{in0} + \lambda_0 C_0\right)/K_{si}\,. \tag{3.8}$$

The right part of an inequality is expressed through library parameters and load, and this ratio can be used for computation of conditions (3.2). If the primary element is one-cascade then the inequality (3.8) becomes simpler.

In many cases it may appear reasonable to use the overestimated but more simple in calculations requirements rather than conditions (3.2). Let's require to meet the following conditions that guarantees (3.2) as well:

$$2\tau_{t0}/K_{si} < \tau_{li}, \quad i = 1,2,\ldots, \tag{3.9}$$

where τ_{li} is the threshold delay of i-th secondary element.

It depends on the thresholds of elements u_{li}^* attached to the output of i-th element that will be taken into consideration by the coefficient α_i: $u_{tpi}^* = \alpha_i \Delta U$,

$$\tau_{li} = 2\tau_{ti}\alpha_i, \quad i = 1,2,\ldots, \tag{3.10}$$

where τ_{ti} is the propagation delay of i-th element; α_i is in the range $0.25 \div 0.75$ for real elements.

Finally, the strengthened requirements accept the form:

$$\tau_{in0} + \lambda_0 C_0 < K_{si}\alpha_i\left(\tau_{ini} + \lambda_i C_i\right), \quad i = 1,2,\ldots. \tag{3.11}$$

The inequalities (3.11) guarantee applicability of the Muller's model to an element connection in the case of absent transport delays in wires.

3.4 Base elements of S^3 circuits

In [2] and other related publications the question is investigated on possibility of a self-timed representation in NAND and NOR logical bases. Other approaches to the question of the self-timed basis was not represented in the literature.

As opposed to those publications, in this subsection a physical basis of the S^3 circuits is considered, i.e. real elements that are possible to implement correctly the logical model of an S^3 circuit. Also as opposed to those publications, we shall understand here under the word *element* a real element having own electrical circuit and implementing some logic function.

The necessity of studying the physical basis is caused by following reasons. Specificity of the S^3 circuit design has a troublesome consequence: if any logical function can not be implemented as a single element, the description of the circuit or its fragment has to be reconstructed from the very beginning. Therefore, knowledge of the implementation basis is necessary already on the stage of logic design. In particular, the problem of S^3 circuit implementation basis arises when custom or semicustom element libraries are used. An important question is selection of one implementation variant from several possible when a logic function can be implemented in various ways, with different electrical circuits. At last, the development of perspective element libraries for the S^3 circuits is impossible without a research on their physical basis.

In the theory of the S^3 circuits, each element is defined by an equation of the Muller's model [8]. It is obvious that, if a real element is described by one equation of this model, it can be used as an S^3 base element.

Thus, the task consists in:

- establishing a correspondence between characteristics of the electrical element circuits and concepts of the discrete Muller's model
- selecting a set of S^3 base elements on this basis.

3.4.1 Structure and characteristics of S^3 base elements

Properties of the Muller's model dictate following conditions for its elements:

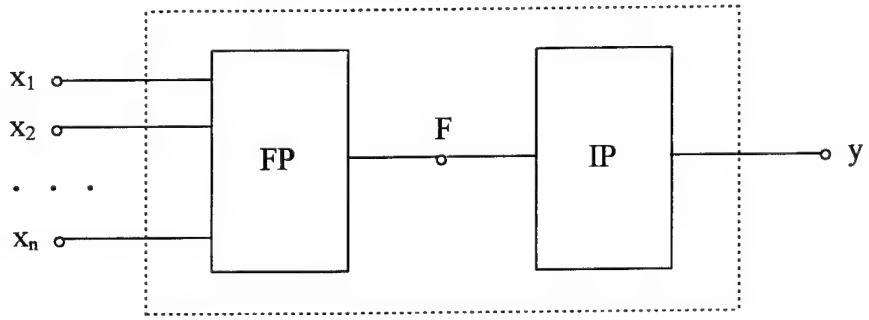
- each element has one output and one or several inputs
- signals on all inputs and outputs accept only values of logical 0 or 1
- element delays (excitation durations) are reduced to the element output.

Let's remind that the term *excitation* means by Muller an internal process starting since a change of one of its inputs and resulting in a change of its output.

Let's further name *logic* elements that have only inputs and outputs, with both inputs and outputs accepting only values of 0 and 1. The elements having bi-directional inputs-outputs, so called *tri-state* elements, with outputs accepting a special high-impedance state or other nonbinary states, are not considered as logic elements and do not satisfy two first requirements of the Muller's model.

A typical structure of an S^3 base element not violating the Muller's model is represented in *fig. 3.6*.

If the FP delay will be concentrated on its output, the condition 3 of the Muller's model will be satisfied. For this, we shall require that the FP were implemented as a one-cascade circuit with regard to all inputs. This requirement can be formulated more exactly as follows.



FP — a multi-input functional portion;
IP — a portion containing arbitrary number of sequentially connected inverters.

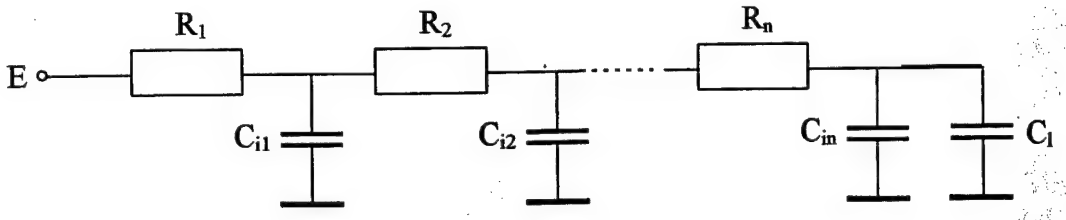
Fig. 3.6 **Typical Structure Of an S^3 Base Element**

For any input x_i $(i = 1,\ldots,n)$, following condition must be satisfied: if x_i excites the element then all switching current determining the change of potential in the point **F**, fig. 3.6, must pass through one of transistors which gate is connected to the input x_i.

Input circuits of many library elements meet this principle. For us now important is solely that only inverters would follow the one-cascade input portion.

Let's consider fulfilling the condition 3 of the Muller's model. It concerns the speed of one-cascade CMOS elements. The delay for such elements is determined by the load recharge

through a chain of sequentially connected transistors in accordance with the equivalent scheme in *fig. 3.7*.



R_i — resistance of an open transistor (nonlinear);
C_{ii} — internal (parasitic) capacitance of a node, i = 1,..., n ;
C_l — load capacitance;
E — power supply.

Fig. 3.7 **Equivalent Scheme Of the Load Recharge**

Each of internal capacitances C_{ii} is a sum of parasitic capacitances of the gate, source, drain, and internal wire connecting transistors. The nonlinearity of R_i performs function of current limitation.

The delay in such a nonlinear circuitry is the sum of delays of all nodes:

$$t_d = \sum_{i=1}^{n} t_{di} = \sum_{i=1}^{n}\left(\sum_{k=1}^{i} R_k\right) C_{ii} + \sum_{i=1}^{n} R_i C_l \,. \qquad (3.12)$$

It is seen from (3.12) that the delay consists of two parts — internal and load delays:

$$t_d = t_{in} + t_l \,. \qquad (3.13)$$

Let's assume all circuitry resistances and node capacitances to be identical, i.e. $R_i = R$, $C_{ii} = C_i$ $(i = 1,\ldots,n)$. Then:

$$t_d = nRC_l + RC_i \sum_{i=1}^{n} i \,. \qquad (3.14)$$

Let's name the maximum number of consecutively connected transistors in a load recharge circuitry as *delay index* (DI) of the element.

Let's designate $\tau_l = RC_l$, I — delay index, $\alpha = C_{ii}/C_l$. In this case $I = n$. In the accepted notation, the maximum element delay will be defined as follows:

$$t_{d\max} = \tau_l\left(I + \frac{I(I+1)}{2}\alpha\right). \qquad (3.15)$$

It may be seen that the internal delay is square proportional to the element DI.

Depending on the amount of internal transistors and lengths of internal wires, the coefficient α can accept values from some tenths to some hundredths. The formula (3.15) shows that the internal delay begins to play a noticeable role already at DI of a few units.

The last conclusions made in supposition of equality of resistances and parasitic capacitances remain valid in a general case.

Thus, the maximum delay of an one-cascade element grows versus DI increase at first linearly (until the second member in (3.15) begins to contribute noticeably) and then on the quadratic law. It is necessary to define advisable limitation of DI for base S^3 elements.

It is known that any logic function with the number of arguments more than two can be implemented (with accuracy to inversion) by one or several cascades of CMOS elements, *fig. 3.8*.

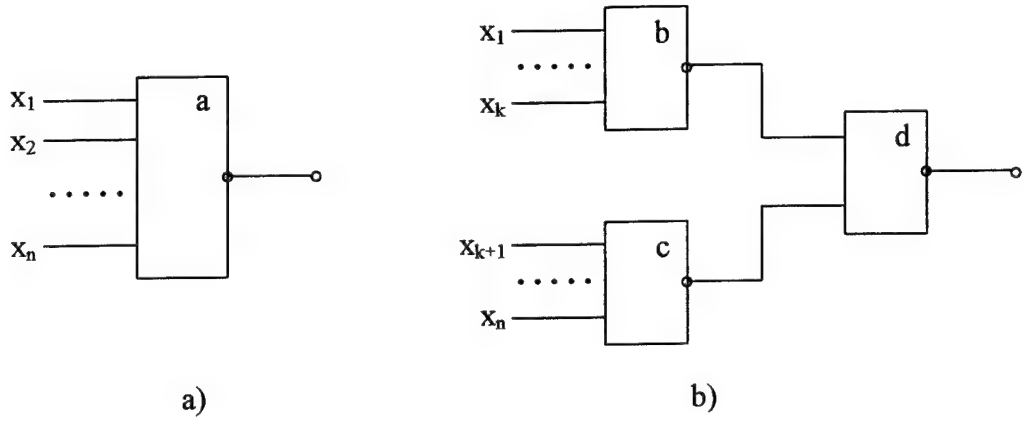


Fig. 3.8 **One- (a) And Two- (b) Cascade Implementation**

The total delay of the two-cascade circuit will be determined by the DI sum of 1st and 2nd cascades, that is:

$$I_\Sigma = \max(I_b, I_c) + I_d . \tag{3.16}$$

Simultaneously, the following ratio should be met:

$$I_b + I_c \geq I_a , \tag{3.17}$$

with the equality being more probable.

Several first values I_a are given in *table 3.1*.

First DI values				*Table 3.1*
I_a	I_b	I_c	I_d	I_Σ
3	2	1	2	4
4	2	2	2	4
5	3	2	2	5
6	3	3	2	5
7	4	3	2	6
8	4	4	2	6

Comparison of the first and last columns of the table shows that since $I_a = 6$ the two-cascade implementations are faster than one-cascade even without respect to internal delays. With internal delays, the one-cascade implementations lose in the speed already at $I_a = 5$. Hence, the elements with $DI > 4$ are ineffective from the viewpoint of speed and practically not used in existing libraries of VLSI manufacturers.

Being efficient, S^3 base elements can't help being restricted by this general limitation on the DIs. Additionally, we shall require that one more relation were satisfied:

$$t_{in} << t_l . \qquad (3.18)$$

In this case, the element delay is determined by its load, and the condition 3 of the Muller's model is fulfilled.

Meeting the inequality (3.18) depends on the DI, element load, number of its inputs, structure of transistors interconnections, topological and technological parameters. In order to meet it at small loads, some extra limitations are to be introduced on the number of inputs and structure of interconnections. Such limitations are rather easy calculated for any known integrated circuit manufacturing technology.

Thus, a set of S^3 base elements is defined by the following conditions:

- the structure of each element should correspond to *fig. 3.6*, where FP is a one-cascade transistor circuitry, and IP is a chain of inverters
- values on the element inputs and outputs are solely logic 0 and 1
- FP DI is not greater than 4
- the number of inputs and structure of interconnections should be such that the delay would be defined mainly by the load.

3.4.2 Kinds and functions of the S^3 base elements

Derivative from the definition of a set of S^3 base elements given above, actual become the questions regarding their existence, total quantity, and to which extent do they restrict possibilities of circuit implementation.

To answer these questions, we have to find a relation between the base element limitations and parameters of executed functions. Functions will be represented in the *minimum disjunctive normal form* (MDNF).

The functions associated with the S^3 base elements may be only *isotonic* or *antitonic*. The first, presented in the MDNF, have no negations of arguments, the latters have negations of arguments only. If the IP in *fig. 3.6* has odd number of inverters then the element function will be isotonic, if it has even number of inverters, no inverters included, then the function will be antitonic.

To begin with, we shall consider elements without inverters, i.e. containing an FP only. Their structure in transistors is represented as a multi-input circuit of the *push-pull* type, *fig. 3.9*.

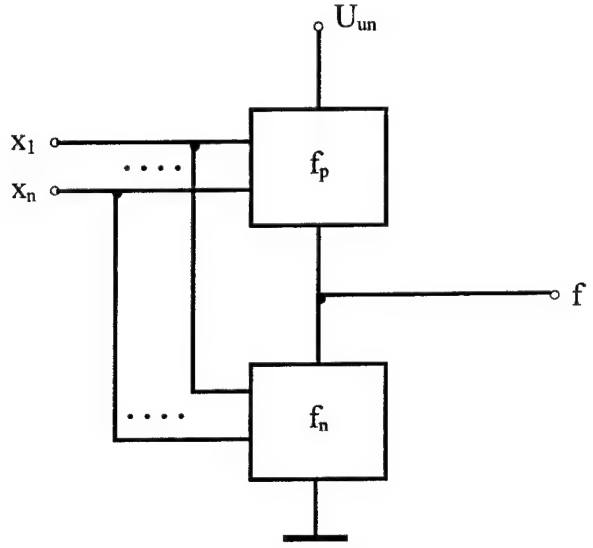


Fig. 3.9 **General Structure Of the S^3 base element FP**

In the *fig. 3.9* and further, indices *p* and *n* designate circuitries containing the p- and n-transistors, respectively.

For description of transistor structures of such CMOS elements, it is convenient to use symbolic functions $\mathbf{f}_p$ and $\mathbf{f}_n$. The symbolic function is an isotonic logic function representing the

transistor connections, with logic AND meaning their sequential connection and logic OR — parallel.

Relation between an element output function and symbolic functions is given by the expression:

$$f = f_n,\ f_p = f_n^*, \tag{3.19}$$

where the asterisk means a *dual* logic function.

Representing functions f_p and f_n in the MDNF enables to specify easily their relation to limitations on the base elements. The element DI will be equal to the maximum number of variables in the implicants (terms) of both functions. In first approximation, it is permissible to consider the internal parasitic delay to be proportional to the number of implicants, in the MDNF. For any given technology and topological structure of elements, it is possible to find an appropriate limitation on the number of implicants.

Since the functions f_p and f_n are unambiguously mutually defined by the equation (3.19), sufficient will be any of them having maximal parameters: the DI and number of implicants. Such a function meeting also the limitation on the DI is named *base function*. Thus, base function is an isotonic logic function, in the MDNF, with a limited number of implicants, and meeting the limitation on the DI.

Each base function produces four S^3 base elements functions:

- isotonic, basic
- isotonic, dual to basic
- antitonic, inverse to basic
- antitonic, inverse to dual.

A computerized search for base functions has found their number being rather great. The number of the found functions is about 20,000; while their total number is still several times more. Following tables are able to give a notion of the number of base functions.

Note. Designations used in the tables:
A — number of arguments;
T — number of terms (implicants);
x — functions presumably exist but are not found yet.

Base Functions with DI = 2			*Table 3.2*
T	*A*		
	2	3	4
1	1		
2		1	1
3		1	1

Base Functions with DI = 3								*Table 3.3*
T	*A*							
	3	4	5	6	7	8	9	10
1	1							
2		3	2	1				
3		3	13	13	7	2	1	
4		2	26	82	81	33	5	
5			24	267	434	200	30	1
6			10	490	1625	713	x	x
7			4	717	3130	x	x	x
8			1	816	x	x	x	x
9				496	x	x	x	x
10			1	697	x	x	x	x

Base Functions with DI = 4									*Table 3.4*
T	*A*								
	4	5	6	7	8	9	10	11	12
1	1								
2		4	3	2	1				
3		7	37	47	38	17	7	2	1
4		9	177	613	982	x	x	x	x
5		6	590	x	x	x	x	x	x
6		1	1367	x	x	x	x	x	x
7		1	2224	x	x	x	x	x	x
8			2616	x	x	x	x	x	x

As an example, all base functions with DI = 2:

$$\begin{aligned} F_1 &= AB, \\ F_2 &= AB \vee AC, \\ F_3 &= AB \vee AC \vee BC, \\ F_4 &= AB \vee CD, \\ F_5 &= AB \vee AC \vee BD. \end{aligned} \tag{3.20}$$

In particular, function F_2 produces following functions of S^3 base elements:

$$\begin{aligned} f_1 &= ab \vee ac, \\ f_2 &= a \vee bc, \\ f_3 &= \bar{a}\bar{b} \vee \bar{a}\,\bar{c}, \\ f_4 &= \bar{a} \vee \bar{b}\,\bar{c}. \end{aligned} \tag{3.21}$$

The given consideration shows the logical (and also physical) basis of the S^3 circuits is rather wide and practically does not restrict implementation possibilities.

3.4.3 Use of the base elements in S^3 circuits implementation

It is known that any circuit can be implemented by the simplest elements 2-NAND or 2-NOR though such an implementation assumes too large expenditures of transistors. An

opposite extreme solution, with very complicated elements, allows to save transistors but is not effective on the speed. The DI and number of implicants in the element function serve as a bounding factor here.

The base elements were defined already in respect to these limitations. Therefore, implementation of S^3 circuits on the S^3 base elements is expected to be most optimal, and the more complex base elements will be used the more transistors saving can be achieved.

The existing elements libraries of the VLSI manufacturers can be applied to implementation of the S^3 circuits. However, two important remarks should be made.

The available libraries contain from tens to hundreds of elements, part of which meet limitations on the S^3 base elements. The total number of the S^3 base elements as it has been shown above achieves tens and hundreds of thousands. Thus, no existing fixed VLSI libraries can comprise all base elements. For example, even such simple elements which correspond to the base function F_5 in (3.20) were not found in any of ten investigated particular VLSI libraries. Besides, the base elements included in the existing libraries are most simple that increases the average expenditure in transistors.

All this permits to make a conclusion that, though the S^3 circuits can be practically implemented on base elements of any existing fixed VLSI library, such an implementation, at the same other conditions, will require more expenditures in transistors as compared with the implementation on a specific full set of the S^3 base elements.

As a result, it becomes evident that an optimal strategy of the S^3 circuits implementation consists in generation of necessary base elements for each circuit being designed. Existing possibilities of the VLSI design and manufacturing enable to do it without considerable difficulties.

3.5 Conclusions

1. The strictly self-synchronous circuits are naturally implementable in the CMOS technology.
2. Explicit S^3 conditions for circuits with branchings are obtained. They are satisfied in most of practical cases if the difference of transport delay values after branchings is relatively small.
3. A class of the S^3 base elements in the CMOS technology is defined. The elements enable correct and efficient implementations of the S^3 circuits.

4. The class of the S^3 base elements is rather wide and includes tens and hundreds thousands of elements. No existing fixed library of a VLSI manufacturer is able to encompass this class entirely.
5. The S^3 circuits can be implemented on elements of existing VLSI libraries (as containing a part of most simple S^3 base elements). Such implementations, however, will be more expensive in comparison with the implementation on a full set of special S^3 base elements.
6. The most optimal strategy of implementation of the S^3 circuits is generation of necessary S^3 base elements for each specific circuit.

4 CAD TOOLS FOR THE S^3 CIRCUIT DESIGN

Works associated with the CAD tools for the S^3 integrated circuits design always were paid a considerable attention within the whole IPI RAN's program on the self-synchronization. They comprised both theoretical researches and practical developments. The latters resulted in two experimental CAD systems for the S^3 circuits design support:

- medium level CAD system *FORCAGE* for the gate and block levels
- low level CAD-through system *RONIS* covering all levels, from logic (in future it is supposed to replace the FORCAGE system) to topology.

The CAD system FORCAGE, initially proposed as experimental, approached, in its last version *FORCAGE 3.0*, the status of a functionally complete product. The CAD system RONIS is now in an initial stage. Accordingly, this report is concerned only with the FORCAGE system facilities. The RONIS system will be discussed in a next report.

4.1 General information about the FORCAGE system

4.1.1 Theoretical basis

The specific nature of the S^3 circuits requires a rather high skill for the "manual" design. This drawback, in comparison with the conventional synchronous circuits design, is a natural "counterweight" to advantages of the S^3 circuits. Since no modern VLSI designs are possible without powerful CAD systems, this drawback concerns CAD tool creators rather than VLSI designers.

At first glance, problems of the S^3 circuit design seem different from those of conventional, synchronous for example, not so much that it would make impossible to use existing VLSI CAD systems. For example, if an initial specification simulating any specific process is given, it is analyzed by the CAD tools on specific S^3 features of the process. In case of detection of S^3 incorrespondences, the initial specification is updated to remove them, and an S^3 circuit is synthesized in compliance with the updated specification. However, it has been shown in section 3 that cost of updates will be excessive in regard to the S^3 circuits. Specificity of the S^3 circuits makes to refuse from conventional approaches and concepts on all design stages and to develop special S^3-oriented CAD tools.

In the synchronous schemotechnique, models are based on an explicit timing framework, i.e. it is *deterministic*, and any circuitry has to comply with this framework with respect to dynamic parameters of all consisting elements. The self-synchronous schemotechnique does not have the notion of timing, it is *behavioristic* on its nature. Behavioristic models become a basic tool for the S^3 circuits design that assumes more complicated algorithms of logical analysis and synthesis.

At present, theoretically valid algorithms of behavioristic analysis and synthesis proper to the S^3 schemotechnique have been developed and partially implemented in the CAD system FORCAGE.

4.1.2 Basic features

The FORCAGE system was developed in the MS DOS environment, with the RAM extended to 64 Mbytes. It consists now of 9 subsystems supporting following functions:

- editing — multi-window editor *Edit*
- S^3 circuits analysis — subsystem *TRANAL 3.0* (at present, a more powerful and fast subsystem *BTRAN* is under development
- analysis of transition diagrams — subsystem *TRANDI* (integrated into the subsystem TRASYN and not available in FORCAGE 3.0 as a separate menu item)
- circuit minimization — subsystem *TRAMIN 2.0*
- S^3 circuits synthesis — subsystem *TRASYN 3.0*
- conversion from a block presentation form to a linear one — block transformer *TRABT* (integrated into the subsystem TRANAL 3.0 and not available in FORCAGE 3.0 as a separate menu item)
- conversion of graphical images of the PC-CAPS editor to input languages of appropriate subsystems — subsystem *TRACON 1.1*
- conversion of a VHDL description to an internal circuit representation language — *TRAVHD 1.3* (i.e. a VHDL to FORCAGE converter)
- FORCAGE demo-roll — *DEMOROLL 2.0* (slide-film)
- demo of the design process *DEMOSYS 2.0* (autonomous FORCAGE demo system).

The FORCAGE system is compatible with some PCAD 4.50 utilities: PDIF-OUT, PDIF-IN, PC-COMP, PC-LIB, PC-PRINT, PC-LINK — that enables to input graphic images of objects designed in PCAD, e.g. library elements.

Each subsystem looks like an integrated shell that can be divided in two parts: a user interface and a kernel, with its contents depending on assignment of the subsystem.

The FORCAGE system possess following general features.

It is oriented on the S^3 schemotechnique and CMOS technology that assumes (see section 3) to meet some restrictions. For example, the subsystem TRANAL is responsible for correctness of combination coefficients for inputs and branching coefficients for outputs. (These restrictions are not conceptual and may be changed by user in a dialog mode.)

Another much more inherent restriction on the FORCAGE subsystems is maximum dimension (i.e. complexity) of the processed S^3 circuit: the number of variables in a circuit is limited by 256. Analysis of more complicated circuits, with the number of variables exceeding 256 and the number of parallel processes being large, requires too much computational resources. Since the TRANAL subsystem uses a model in global states, transition diagrams (TD), circuit descriptions augment exponentially versus the number of variables. (In the FORCAGE system, on the gate representation level, logic inputs and outputs of zeroth level elements of a circuit are considered as variables.)

Note. Complexity of S^3 circuits, with the number of variables below 256, is expressed in the number of CMOS transistors needed for its implementation. For example, complexity of regular circuits with only few inputs (registers, FIFO-type buffers, etc.) having less than 256 variables can be up to 10,000 transistors.

The maximum number of variables in the FORCAGE system version is determined by the RAM capacity available for saving data structures. Some of these structures are indivisible and have to be represented in RAM as continuous units. From algorithmical point of view there are no restrictions on the number of variables. But the methods of program allocations under MS DOS rather quickly exhaust existing RAM resources. At the real RAM capacities, the TRANAL subsystem not always permits to process even declared 256 variables. The BTRAN subsystem eliminates this shortcoming and provides increasing the number of variables proportionally to the accessible memory capacity.

The analysis and synthesis subsystems in the FORCAGE system support design of closed-type circuits that constrains a user to simulate its environment. For the S^3 circuits implementing an automata model, a delay element or inverter may represent the environment. For combination circuits with branchings, simulation of the environment is a nontrivial task that may require even more complicated environment than the analyzed circuit itself.

Note. Design of S^3 circuits of large dimensions can be implemented by FORCAGE 3.0 after preliminary division in a number of more simple S^3 sub-circuits (block decomposition) and their subsequent combining by the designer, "manually" using known standard methods, in a whole S^3 circuit (block composition or synthesis). Unfortunately, there exist probability to bring in errors while passing from local (within a block) transition process completion indicators to a global (within the whole S^3 circuit) indicator.

The system provides possibility to design correct S^3 circuits on the logic level. To preserve S^3 features of a circuit certified on the logic level when passing to the transistor-topological level, it is necessary to develop appropriate subsystems.

The FORCAGE system is applicable to conventional CMOS technologies. Transition to sub-micron technologies needs development of an appropriate theory of S^3 circuits design that would be invariant to delays in wires and transistors. At present, only some fragments of such a theory have been created.

4.2 Brief overview of the subsystems

4.2.1 TRANAL subsystem of S^3 circuits analysis

The TRANAL subsystem is designated for checking correctness (i.e. independence of behavior from elements delays) and calculating behavioristic characteristics of digital circuits given by Muller diagrams.

A circuit is defined as a set of logic elements $\{Z_1,..., Z_n\}$ in which every input of a logic element is connected with one output and no two outputs are connected with each other. The state of the circuit in any moment is a set of signal values in its nodes, i.e. outputs of binary logic elements. Inputs of the circuit can be also considered as logic elements without inputs (or generators of zeros and ones).

The existing theoretical and practical results in the S^3 schemotechnique are based on the Muller's model, with the following assumptions on the delay nature:

a) an element of a circuit possesses an inertial delay of an arbitrary but finite value;
b) deskew of delays in wires after branchings is negligible in comparison with delays of elements;
c) own delay of an elements and the delay on its output wire before branching are considered as brought to (reduced to) the element output.

An element Z_i satisfying such assumptions can be represented as a sequential connection of a noninertial logic element F_i and an inertial delay D_i of an arbitrary but finite value. In principle, in wide range of application areas, such a hypothesis is valid.

The circuit operation is described by a procedure of its transition from one state to another. Reflection of the circuit operation in a visual form defined by the Muller's model is TDs. TD is an oriented graph, with its vertices corresponding to circuit states. One of the most important consequence of theory of S^3 circuits consists in a proof of the fact that, if such a circuit operates properly, it is sufficient to be convinced in *semimodularity* of corresponding TDs with respect to some initial state.

In the TRANAL subsystem, provision was made for conversion of a user description of a circuit from the form of a system of Boolean equations into an internal representation of a corresponding TD. The TDs investigations in the analysis subsystem allows to make a conclusion about its correctness on S^3 criteria.

The result of the analysis consists in fixing semimodularity of a given circuit with respect to a given initial state or detecting either violation of semimodularity with indication of a conflict state and conflicting variables or a *blind* state.

The restrictions and characteristics of the TRANAL 3.0 subsystem:

- number of variables — not more than 256
- combination coefficient on inputs — from 1 to 8 (user definable)
- representation of logic elements — in both disjunctive and conjunctive normal forms
- needed RAM capacity — not less than 200 K bytes
- possibility to assign types of spacer used and to check the circuit correctness on the set of input data sets within one simulation cycle.

4.2.2 TRAMIN subsystem of minimization of Boolean functions

The TRAMIN subsystem of minimization of partially defined Boolean functions is designated for construction of minimal characteristic equations of logic elements of a circuit.

Realization of the subsystem is based on a theoretical conclusion about possibility to pass from a semimodular TD to a system of characteristic functions of the system elements. Strategy of searching for element characteristic functions that has been chosen in the subsystem enables to obtain a good level of minimization in acceptable time for various classes of Boolean functions: arbitrary disjunctive, monotonous disjunctive, and bracket forms.

An important feature of TDs for most part of circuits that are widely met in practice is a comparatively low percentage of states included into a work cycle as compared with the total possible number of such states. That is why the procedure of synthesis of element characteristic

functions has to provide a given operation only within the work cycle; characteristic functions on states that are not in the work cycle can be defined arbitrarily. Thus, in the task of search for a minimal representation of element characteristic functions, a search of a minimal representation of weakly-defined Boolean functions takes place.

The result of minimization is an approach (dependent on the user controlled amount of look-overs) to a minimal disjunctive form, arbitrary or monotonous (defined additionally).

Restrictions and characteristics of the TRAMIN 2.0 subsystem:

- number of variables — from 4 to 16
- number of vectors — from 4 to 4000
- number of considered minimization variants — from 1 to 9999
- number of alternative minimization results — from 1 to 10
- needed RAM capacity — not less than 250 K bytes.

4.2.3 TRASYN subsystem of S^3 circuits synthesis

The TRASYN subsystem is designated for synthesis of S^3 circuits, i.e. obtaining a system of Boolean equations of logic elements from an initial behavior description of circuit operation.

The subsystem provides:

- analysis of the initial behavior description given in a language of *change diagrams* (CDs)
- synthesis (if possible) of Boolean equations of logic elements corresponding to the initial CD
- minimization of Boolean equations.

The TRANAL subsystem mentioned above in subsection 4.1.2 specifies an input representation, in a TD form, of an S^3 circuit being designed. Such a model representation, referred to as *states model*, provides a detailed description of the circuit operation. In each specific moment and while passing from one state to another, the circuit is characterized by complete information about states on all of its outputs.

TDs meet requirements of exact circuit description rather well and are a widely used means for the model representation. A lot of S^3 circuits analysis and synthesis tasks have been solved for it. But even for simple elements (library elements of lowest levels), "visuality" of description of their operation is not sufficient. The operation description length increases exponentially versus the number of elements, and, with the circuit complexity growing, the advantage of the TD description completeness becomes a disadvantage.

One of ways to shorten the description is replacement of the states model by a *behavioristic model* in which, at any moment, of interest are states not of all circuit components but only of those that are being changed this moment.

The TRASYN subsystem uses a behavioristic form of the circuit representation, namely the CD form, with signal changes being considered as events in a circuit. It includes modules for both analysis and synthesis. The analysis module is designated to check correctness of a process specification given by a CD. The outcome is either statement about correctness of the initial description or localization and classification of incorrectness points. To obtain the system of equations on the synthesis stage, reduction to TD global states from CD is performed. The possibility of such a procedure is based on the equivalence between both classes of correct CDs and semimodular TDs.

The condition for existence of a system of Boolean functions implementing a given TD is absence on the circuit inputs of contradicting states, i.e. states with the same Boolean values but different post-history. In case of contradictions, the subsystem indicates contradicting states and information about process implementation paths that has brought the circuit to the contradicting situation.

Main difficulties in synthesis are connected with the exponential growth of description complexity during constructing TDs from CDs. The synthesis outcome consists in either successful construction of a TD or indication of a point where a real or imaginary contradiction has been found.

Main characteristics and possibilities for user's control on the minimization stage are the same for the TRASYN subsystem as those for the TRAMIN subsystem. The main difference of the TRASYN 3.0 minimization module from the TRAMIN module is the increased resolution of the first one: minimization of functions depending on 64 variables is possible.

The subsystem TRASYN is well adapted to the S^3 circuits design-through: from the specification and analysis up to the direct translation into a circuit.

Main characteristics of the TRASYN 3.0 subsystem:

- needed RAM capacity — not less than 350 K bytes
- number of variables in the circuit during analysis — not more than 256
- number of variables in the circuit during synthesis — not more than 256
- number of variables in the circuit during minimization — not more than 64 / 32
- possibility of synthesis of arbitrary number of variables
- possibility of synthesis on a partial description of a circuit.

4.2.4 TRACON subsystem of graphical input

In the subsystems for analysis and synthesis that were described above, special circuit description languages for hardware designers were used: Muller's model represented in the form of a system of Boolean equations in the TRANAL subsystem and CDs, in a list form of adjacent vertices, in the TRASYN subsystem.

In the FORCAGE 3.0 system, graphical means have been involved (based on PC-CAPS editor from the CAD system PCAD) that permits a designer to describe objects, digital circuits and CDs, by their graphical images. Such a conventional for the circuit designers method of representation possesses much higher clearness and is better protected from errors during the input.

For automated conversion of graphical images of objects obtained by the PC-CAPS editor into the necessary specifications on input languages of the TRANAL and TRASYN subsystems, the TRACON subsystem is provided as a unified graphical interface. Such an additional graphical user interface does not exclude possibility to specify digital circuits by a system of Boolean equations (Muller's models) and CDs (a vertix adjacencies list).

The interface FORCAGE / PCAD allows using, under a design process, not only the analysis and synthesis methods of the FORCAGE system implemented in the TRANAL and TRASYN subsystems but also all tools for digital circuits design proper to the PCAD and other compatible systems for automated design of conventional synchronous circuits.

To create descriptions of digital circuits, logic elements of any other libraries of the PCAD system may be applied. In such case, it is necessary to supplement description of a logic element with the Boolean equation written in respect to contacts names and in correspondence to requirements on the input language of the TRANAL subsystem.

4.2.5 System of S^3 element libraries on logic and behavioristic levels

At present, a hierarchical library of logic elements and CDs designated for automated design of S^3 circuits in the CMOS technology is created. Depending on purposes and forms of initial representation of S^3 elements, diverse variants of S^3 elements implementation are provided and supported.

When only the TRANAL subsystem is used for analysis of logical circuits represented in the form of Boolean equations, a library of S^3 modules is formed as a specifications set in the

TRANAL input language (*spec. LSTM TRANAL*). When the TRAMIN subsystem is used, a descriptions of S^3 modules is formed in the TRAMIN input language (*library LSTM TRAMIN*). In the latter case, the term *S^3 modules* is valid only after certification of results of minimization by the TRANAL subsystem as the TRAMIN subsystem does not guarantee S^3 properties of the implementation. Libraries are produced not involving the graphical forms of S^3 circuits representations.

Nevertheless, namely the graphical form is a most convenient representation for the S^3 modules as proper to the editor PC-CAPS. Automated translation, by the TRACON subsystem, of a graphical description into a TRANAL input description allows a conventional user to consider that form of representation as basic. That is why the library of S^3 modules in the form of logic elements LSTM TRACON.LE is most complete, multi-level, and, to a large extent, corresponds to the notion "library".

As a supplement to the library LSTM TRACON.LE, descriptions of logic elements from libraries of the PCAD and other CAD IC systems using the PDIF format may be applied. In such cases, there is a limitation on the pin number: not more than 255 pins per element. Besides, the description of a logic element has to be supplemented by a Boolean equation written with respect to the pin names (in compliance with the requirements to the TRANAL input language). For automation of this process, it would be reasonable to create a special converter PDIF / TRACON.

For support of design on the behavioristic level in the FORCAGE 3.0 system, two libraries have been created:

- of CDs of zeroth level (*LCD-0*) having 57 elements
- of standard cells (*LSS*) having 15 elements.

4.3 Directions of development of the FORCAGE 3.0 system

The FORCAGE system is supposed to be developed in several directions.

1. It is necessary to develop an improved revision of the system that would not depend, regarding the permissible number of variables, on memory limitations.
2. Change diagrams should be implemented on the stage of analysis in a new BTRAN_CD subsystem. That will enable to process circuits with hundreds of gates (complexity of verification algorithms is estimated by the polynomial, for CDs, rather than exponential, for TDs, law versus the size of a description). On one hand, it solves the problems of memory capacity and enhancement of system performance. On other hand, it enables to study

approaches to integration of the FORCAGE system with industrial CAD systems of upper (system) and lower (topological) design levels.

3. To guarantee full S^3 circuits design cycle and closer integration with the industrial CAD systems, it is desirable, in addition to S^3 circuits synthesis on zeroth level element basis, to implement synthesis in a basis of existing libraries, i.e. to work out a subsystem covering necessary functions by elements of existing libraries.
4. In the near future, effort of the FORCAGE system designers should be concentrated on the automated design-through of the functionally full S^3 circuits set on the logic level:
   - open S^3 circuits
   - S^3 circuits implementing branching algorithms
   - arbitrating circuits (S^3 arbiters)
   - circuits with "wired ORs",

which implies being necessary:

- to improve characteristics and provide full compatibility of the existing subsystems within the FORCAGE system.
- to provide support of VLSI design for much higher integration levels by increasing the permissible number of variables at least to 1024.

4.4 Conclusions

1. Based on the original scientific researches, the FORCAGE system surpasses existing noncommercial foreign analysis and simulation systems for self-timed circuits (commercial CAD systems for self-timed circuits, S^3 circuits moreover, still do not exist).
2. The FORCAGE system supports all main stages of logic design (analysis, minimization, and synthesis) of S^3 circuits with the number of variables up to 255 that, for CMOS regular S^3 circuits, approximately corresponds to 10,000 transistors.
3. The system provides design on both gate and behavior levels of circuit representation.
4. The interface FORCAGE / PCAD enables to use in designs not only "own" methods of the FORCAGE system itself but also some design possibilities proper to PCAD and other compatible systems.

LITERATURE

1. Мизин И.А., Филин А.В. Принципиальная база архитектуры естественно-надежных компьютеров // Системы и средства информатики. Вып. 7. - М.: Наука - Физматгиз, 1995, с. 172-198.
2. Автоматное управление асинхронными процессами в ЭВМ и дискретных системах. Под ред. В.И. Варшавского. М.: Наука, 1986, 398 с.
3. Апериодические автоматы / Под ред. В.И.Варшавского. М.: Наука, 1976, 424 с.
4. Изосимов О.А. Методы синтеза и динамического анализа самосинхронных КМДП СБИС. Автореферат диссертации канд. техн. наук. М.: 1994, 15 с.
5. Izosimov O.A., Shagurin I.I. Phisical approach to CMOS module self-timing // Electron. Lett. 1990. Vol. 26, N 22, pp. 1835-1836.
6. Асинхронные интерфейсы. Учебное пособие / [Варшавский В.И., Мараховский В.Б., Розенблюм Л.Я., Яковлев А.В.]. - Л.: ЛЭТИ, 1984, 76 с.
7. Seitz C.L. Self-timed VLSI systems // Proc. Caltech Conf. of VLSI. Pasadena, 1979, hh. 345-355.
8. Muller D.E. Asynchronous logics and applications to information processing. In: Proc Symp. on Application of Switching Theory in Space Technology. Stanford, 1962, p. 289-297.